Estrategia de solución al problema de la anotación de secuencias de ADN mediante la metodología CommonKADS

DANIELA DIAS XAVIER

MÁSTER EN INVESTIGACIÓN EN INFORMÁTICA. FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin Máster en Inteligencia Artificial

Junio 2011

Directores:

Gonzalo Pajares Martinsanz
Federico Morán Abad

Calificación: 10.0, MATRÍCULA DE HONOR

Autorización de difusión

DANIELA DIAS XAVIER

20 de junio de 2011

La abajo firmante, matriculada en el Máster en Investigación en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autora el presente Trabajo Fin de Máster: "Estrategia de solución al problema de la anotación de secuencias de ADN mediante la metodología CommonKADS", realizado durante el curso académico 2010-2011 bajo la dirección de Gonzalo Pajares Martinsanz y con la colaboración externa de dirección de Federico Morán Abad (Departamento de Bioquímica y Biología Molecular, Ciencias Químicas) en el Departamento de Ingeniería del Software e Inteligencia Artificial, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

Resumen en castellano

La Bioinformática ha surgido de la unión entre la Informática y la Biología con el objetivo de tratar el gran volumen de datos biológicos generados en los últimos años. La genómica, uno de los campos de la Bioinformática, es responsable del análisis del genoma, o sea, de las secuencias de ADN que componen el material hereditario de los organismos.

Uno de los grandes desafíos de la genómica es la anotación de genes. Esta tarea consiste en encontrar los genes existentes en una determinada secuencia de ADN para asignarles características biológicas. La anotación de genes debe ser lo más exacta y fiable posible, pues al inferir e inserir una anotación errónea en una base de datos, este error puede ser propagado a futuras anotaciones. Con el objetivo de facilitar este proceso existe una gran variedad de programas y *pipelines* bioinformáticos disponibles.

Con el advenimiento de las nuevas tecnologías de secuenciación, se ha reducido el coste del proceso de secuenciación y, consecuentemente, se ha generado un aumento significativo en el volumen de datos producidos. Analizar estos datos es una necesidad, pero no se puede hacer dependiendo de la intervención humana, debido al cuello de botella generado. Una solución para este problema es desarrollar Sistemas Expertos capaces de anotar automáticamente las secuencias emulando la intervención del especialista en puntos clave del proceso.

Actualmente, los Sistemas Expertos de anotación disponibles fueron desarrollados para anotar genomas completos. Sin embargo, existe una gran demanda por parte de la Comunidad Científica por anotar secuencias de ADN de organismos cuyo genoma completo no ha sido secuenciado. Desarrollar un SE para tal finalidad puede contribuir a la mejora de la calidad de la información a incluir en las Bases de Datos génicas.

La creación de un SE para la anotación de secuencias de ADN no es una tarea fácil debido al gran conocimiento del dominio requerido, así como a la comprensión del razonamiento a aplicar como base de inferencia. Por ello, se hace absolutamente necesaria la creación de un marco general capaz de solventar la problemática antes mencionada. Al tratarse de un problema centrado en la Ingeniería del Conocimiento, la metodología CommonKADS, como estrategia de organización del conocimiento aporta una solución metodológica de relevancia.

En este trabajo se realiza un modelado del problema bajo el paradigma CommonKADS donde se identifica la estrategia a seguir ante un problema de tal envergadura y se establecen las bases metodológicas para abordar el problema de forma general con posibilidades de aplicación a cualquier sistema independientemente de su complejidad. La creación del mencionado marco constituye el objetivo principal del proyecto que se plantea, verificando su validez mediante el diseño de un SE de esta naturaleza.

Palabras clave

Bioinformática, sistema experto, anotación de genes, secuencia de ADN, CommonKADS, extracción del conocimiento

Abstract

Bioinformatics arises from the union of Informatics with Biology aiming to deal with the huge amount of biological data generated in the last years. Genomics, one of the fields of Bioinformatics, is responsible for the analysis of the genome, that is, the DNA sequences that compose the organisms' hereditary material.

One of genomics biggest challenges is gene annotation. This task consist in find the genes that exist within a DNA sequence and assign them biological features. The gene annotation should be as accurate and reliable as possible, because when a missannotation is generated and uploaded into a Database, this error could be propagated to future annotations. In order to facilitate the annotation process a big variety of Bioinformatics programs and pipelines is available.

With the advent of new sequencing technologies the cost of sequencing has decreased, increasing significantly the amount of data produced. Analyze all this data is a necessity, but it is important to avoid the bottle neck created by human intervention during the process. One solution to this problem is to develop Expert Systems that are able to annotate gene automatically and emulate the expert involvement in certain key points of the process.

Currently the Expert Systems for annotation that are available were developed to deal with the whole genome. Nevertheless there is a great demand by the Scientific Community to annotate DNA sequences of organisms whose whole genome has not been sequenced. Creating an Expert System capable of annotating DNA sequences without considering the genome context would contribute to improve the quality of the information to be uploaded into gene Databases.

Developing an Expert System is not an easy job because it requires a huge knowledge of the domain and the understanding of the reasoning process applied as base of inference. Thus, it is really necessary to create a general framework capable of solving the aforementioned problem. As this problem is centered in Knowledge Engineering using the CommonKADS methodology like an organization strategy contributes to a relevant methodological solution.

In this work we model the sequence annotation problem applying the CommonKADS paradigm. Through this methodology it is possible to identify the strategy that best fits the problem mentioned before. Moreover, it establishes methodological bases to tackle the problem in a general way, allowing them to be applied to any problem independently of its complexity. The main goal of the project presented here is to create the mentioned framework, verifying its validity through the design of an Expert System for annotation.

Keywords

Bioinformatics, Expert System, gene annotation, DNA sequence, CommonKADS, knowledge elicitation

Índice general

Agradecimientos

Agradezco inicialmente a mis directores de trabajo final, Gonzalo Pajares y Federico Morán, que me han guiado en los últimos meses con paciencia y conocimiento. Agradezco también a mis amigos y compañeros de trabajo, Arturo Marín y Clara Higuera, que me han apoyado y ayudado cuando más lo necesitaba, y a todos los profesionales de la Biología y de la Bioinformática que han cedido un poco de su tiempo para responder a las entrevistas esenciales para el desarrollo de este trabajo: María Juana Navarro Llorens, Javier Tamames y José Manuel Rodriguez Carrasco. Finalmente, me gustaría expresar mis sinceros agradecimientos a todos aquellos que han contribuido a la ejecución de este trabajo directa o indirectamente.

Dedicatoria

Dedico este trabajo a mis padres que siempre han invertido en mí y me han dado las herramientas necesarias para llegar hasta aquí y, seguramente, poder proseguir; a mi tía querida y única que me da la fuerza necesaria para seguir cuando todo parece oscuro y sin salida; a mi novio, Miguel, que siempre ha sido muy comprensivo y paciente; a mi eterna profesora María Emília que me enseñó el maravilloso mundo de la bioinformática, y, finalmente, a mis compañeros de trabajo (los que están o que ya se fueron) que todos los días entran en el mundo de los ordenadores para abordar los problemas de la biología y sus excepciones.

Capítulo 1

Introducción

Los avances tecnológicos en los últimos años han aumentado considerablemente la cantidad de datos biológicos, especialmente en campos relacionados con la genética y genómica. Estos datos aportan información esencial para la comprensión de las especies y sus mecanismos vitales, incluyendo la especie humana. Junto a esta explosión de datos biológicos ha surgido la necesidad de analizar toda esta información con el fin de extraer el máximo conocimiento en el menor tiempo posible y con alto grado de fiabilidad, tratando de eliminar los problemas derivados del esfuerzo humanano, tarea delegada a la **Bioinformática**.

La Bioinformática es el campo de la ciencia donde la Biología, la Informática y la Tecnología de la Información se unen para formar una única disciplina[30]. Tiene como objetivo almacenar, mantener, analizar, interpretar y relacionar datos biológicos, teniendo aplicaciones en diferentes áreas biológicas, tales como la genómica y proteómica. De forma general, la genómica tiene el objetivo de determinar y analizar la secuencia de ADN integral de un organismo, o sea, su genoma, mientras que la proteómica se encarga de las secuencias de proteínas de todo el organismo, el proteoma.

Uno de los grandes desafíos de la genómica es la **anotación de genes**, que consiste en identificar genes para asignarles características, a partir de una secuencia de ADN. Este proceso requiere alto nivel de conocimiento, pudiendo realizarse de diferentes maneras.

Idealmente, debe ser realizado por expertos que mediante sus conocimientos biológicos y bioquímicos, son capaces de juzgar cuáles son las características correctas que deben ser asignadas a una secuencia dada. Sin embargo, la anotación manual, que es el proceso más fiable y exacto, es lenta y laboriosa.

Con el advenimiento de nuevas técnicas de secuenciación, organismos, e incluso ecosistemas enteros, pueden ser secuenciados en poco tiempo y a bajo costo. Generar datos genómicos, por tanto, ha dejado de ser un problema, no obstante procesarlos, analizarlos y utilizarlos de forma significativa y con utilidad sí que lo es. Debido a esta gran cantidad de datos, cualquier proceso que requiera la intervención humana, en fases tales como procesamiento o análisis, imposibilita o ralentiza la finalización del proceso. Para evitar este cuello de botella, se busca crear métodos *in silico* capaces de manejar los datos genómicos adecuadamente para obtener la información deseada.

La anotación automática, un ejemplo de estos métodos, no requiere un equipo cualificado de profesionales para procesar y analizar las secuencias de ADN, siendo, por tanto, capaz de manipular una gran cantidad de datos en tiempo significativamente inferior al de la anotación manual. No obstante, sus resultados no son tan exactos y fiables como los producidos manualmente, lo que lleva a la necesidad de una verificación ("curado") manual por parte de los anotadores para garantizar su veracidad. Sin embargo, gran parte de la anotación manual producida actualmente es depositada en las Bases de Datos (BD) génicas sin curación previa. Este hecho contribuye a la propagación de errores, puesto que estas BD son utilizadas para inferir otras anotaciones.

La automatización del proceso de anotación puede realizarse a través de *pipelines* de anotación, es decir, de la concatenación de programas bioinformáticos desarrollados para solucionar diferentes problemas de este proceso. Los *pipelines* automáticos poseen las mismas ventajas e inconvenientes de la anotación automática y, consecuentemente, generan datos

menos fiables.

Por otro lado, el empleo de *pipelines* semiautomáticos permite la intervención del experto en puntos clave del proceso, mejorando la calidad de los resultados, pero este enfoque no puede ser empleado para un gran volumen de secuencias. Una vía para solucionar este problema es mediante el uso de técnicas de la Inteligencia Artificial (IA) para "simular" el conocimiento del anotador.

La IA tiene muchas áreas de interés como la robótica, los sistemas artificiales neuronales, la visón artificial, entre otras. El área de los **Sistemas Expertos** (SE) es una aproximación muy exitosa a la solución de los problemas clásicos de IA en la programación de inteligencia[20]. Un SE, o un **Sistema Basado en Conocimiento** (SBC), es un programa que emula la habilidad de tomar decisiones y hacer inferencias de un especialista humano, o sea, un cómputo que se utiliza de un determinado conocimiento para solucionar problemas con una competencia similar a del hombre.

Los SE son capaces de reproducir masiva e indefinidamente la experiencia de uno o varios especialistas a bajo coste. Además, pueden proporcionar respuestas más rápidas, sólidas y complejas, explicando clara y detalladamente el razonamiento que ha conducido al resultado obtenido.

La Figura 1.1 ilustra el concepto básico de un SBC, donde el usuario aporta información al sistema y éste le devuelve una solución basada en su experiencia. Como se puede observar en esta figura, el sistema está formado por dos componentes principales: la **Base de Conocimiento** (BC) y el **Mecanismo de Inferencia** (MI). La BC contiene la información necesaria para llevar a cabo las conclusiones realizadas por el MI. El conocimiento de un SE, es decir, su BC, puede ser representado de varias maneras. Un método común es en forma de **reglas**.

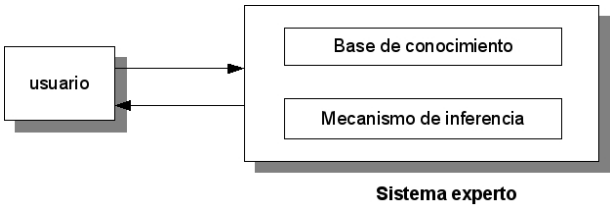

Figura 1.1: *Esquema básico de un Sistema Experto.*

En la Comunidad Científica en Bioinformática existe una inquietud generalizada bajo la creencia de que la creación de un SBC capaz de intervenir en determinados pasos del *pipeline* de anotación podría aumentar la exactitud de este proceso y producir datos con más calidad.

No obstante, crear un SBC es una tarea compleja y laboriosa que requiere el entendimiento profundo del dominio de la aplicación a ser desarrollada, es decir, es necesario comprender cómo el especialista razona al solucionar el problema para poder emular este proceso.

Al proceso de construir un SE se le llama **Ingeniería del Conocimiento** y consiste en la adquisición de conocimiento a partir de un especialista humano o de otra fuente y su codificación en el SE[20]. En la Ingeniería del Conocimiento existen varias técnicas de **adquisición de conocimiento**. La **entrevista**, el método de extracción del conocimiento más empleado, permite la obtención de diferentes tipos de conocimientos, en distintas etapas del desarrollo del sistema, independiente del dominio de aplicación.

Existen diferentes metodologías que soportan el análisis y estructuración del conocimiento para el desarrollo de SBC, entre ellas destaca **CommonKADS**. Esta metodología es flexible y puede ser adaptada a cualquier problema donde se entiende el conocimiento con base en su contexto. Su objetivo consiste en estructurar el proceso de desarrollo de SE y, por tanto, ofrece una serie de herramientas que auxilian en el modelado de tales sistemas.

El objetivo de este trabajo es analizar y estructurar el conocimiento necesario para el desarrollo de un SE para la anotación de secuencias de ADN, sin tener en cuenta el contexto

genómico, mediante la metodología CommonKADS. Este conocimiento será adquirido a través de entrevistas con diferentes especialista en anotación y empleado para crear un esquema general que pueda ser utilizado para cualquier problema de esta índole.

En el Capítulo 2 se presentan los conceptos biológicos y bioinformáticos básicos para la comprensión del problema de la anotación de secuencias de ADN, descrito en el Capítulo 3. El Capítulo 4 explica la metodología CommonKADS y sus modelos empleados para estructurar el conocimiento juntamente con las técnicas de extracción de éste, centrándose en la entrevista. En este capítulo también se describen el análisis y diseño del sistema propuesto. Finalmente, en el Capítulo 5 se presentan las conclusiones llevadas a cabo a lo largo de la investigación realizada. La información adicional al Capítulo 2 se encuentra en el Apéndice A, mientras el Apéndice B contiene los resúmenes de las entrevistas realizadas y el Apéndice C presenta los resultados obtenidos con la implementación propuesta.

Capítulo 2

Fundamentos biológicos y bioinformáticos

2.1. Conceptos biológicos

La **célula** es la unidad mínima de un organismo capaz de actuar de manera autónoma. Está compuesta por estructuras especializadas, denominadas **orgánulos**, que desempeñan diferentes tareas cuyo objetivo es el mantenimiento de la vida de la propia célula.

En la naturaleza existen dos tipos de organismos, los **procariotas**, cuyas células no poseen núcleo y por lo tanto su material genético está flotando libremente en la célula, y los **eucariotas**, que tienen su material genético almacenado dentro del núcleo. Todas las bacterias y arqueas son procariotas, mientras el resto de seres vivos son eucariotas.

La Figura 2.1 muestra la estructura básica de la célula eucariota. Esta célula está compuesta por una **membrana plasmática** que separa el contenido de la célula del exterior y controla el intercambio de substancias entre la parte interna y externa de la misma. Los demás orgánulos, se encuentran dentro de la membrana inmersos en un fluido denominado **citoplasma**, cuya función es ayudar en el transporte de substancias dentro de la célula. El **núcleo** es la estructura que dirige todas las actividades de la célula y es donde está almacenado el material genético. El **retículo endoplasmático** funciona como una red de transporte y un área de almacenaje para substancias de la célula, mientras que el **ribosoma**

produce diferentes tipos de proteínas y el **aparato de Golgi** las empaqueta para almacenarlas o transportarlas al exterior de la célula. Las **mitocondrias** son responsables de la generación de energía en la célula.

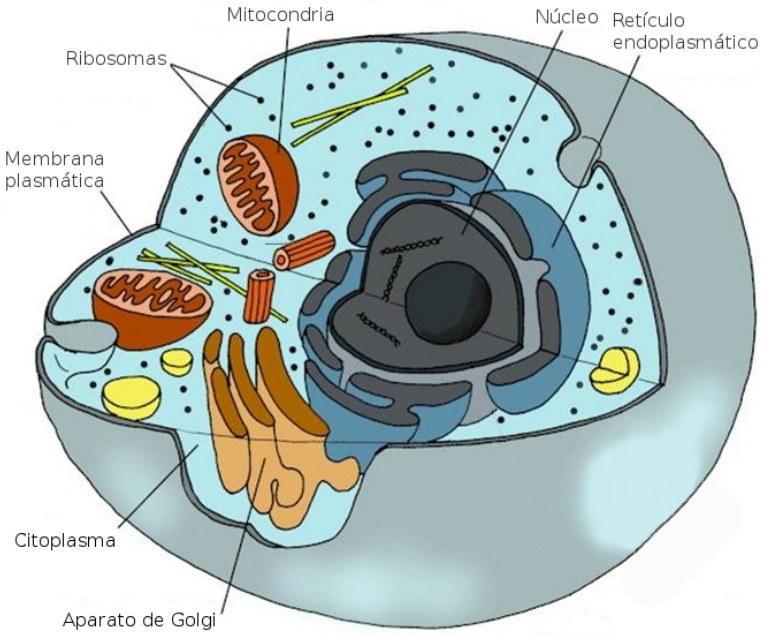

Figura 2.1: *Estructura básica de la célula eucariota.*

La información hereditaria de los seres vivos es almacenada en una macromolécula de ácido desoxirribonucleico llamada **ADN**. Esta molécula consta de dos hebras de nucleótidos que forman una estructura, semejante a una escalera retorcida, llamada doble hélice (Figura 2.2.c). Cada hebra del ADN está compuesta por varios **nucleótidos** que son moléculas formadas por una base que contiene nitrógeno (**base nitrogenada**), un azúcar que contiene cinco moléculas de carbono (pentosa desoxirribosa) y un ácido fosfórico (Figura 2.2.d). Existen cuatro tipos de bases nitrogenadas: adenina (A), guanina (G), citosina (C) y timina (T). En la doble hélice las bases nitrogenadas se emparejan a través de puentes de hidrógeno, dándose la circunstancia de que la A siempre se une a la T y la G a la C. Las bases que se unen entre sí se denominan **complementarias**.

El **ARN** es una macromolécula de ácido ribonucleico cuya composición química se asemeja a del ADN (Figura 2.2.a), a excepción de su pentosa que es una ribosa y de la base nitrogenada timina, reemplazada por el uracilo (U). A diferencia del ADN, su estructura consiste en un única hebra (2.2.b) y no suele formar una doble hélice extensa, aunque bien es cierto que pueden ocurrir pliegues cortos.

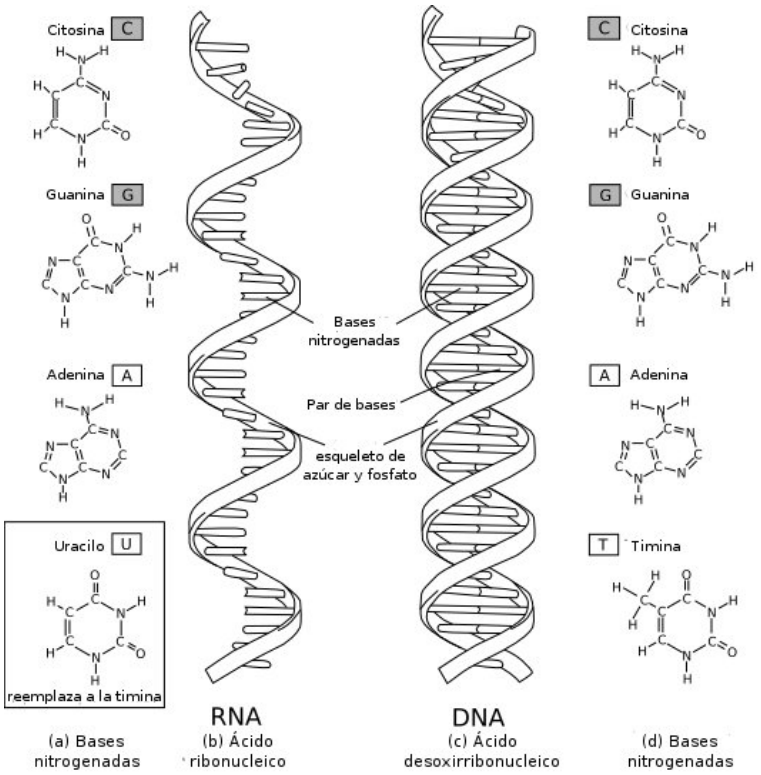

Figura 2.2: *Comparación entre el ARN y el ADN. (a) Composición química de las bases nitrogenadas del ARN. (b) Estructura del ARN. (c) Estructura de la doble hélice del ADN. (d) Composición química de las bases nitrogenadas del ADN.*

Como puede observarse en la Figura 2.3, los **cromosomas** se localizan en el interior de la célula, en el caso de los eucariotas, dentro del núcleo. Están formados por un par de **cromátidas** que se unen en el **centrómero** y son constituidas por ADN asociado a proteínas llamadas **histonas**. Cada organismo posee un determinado número de cromosomas por célula, que puede variar de uno a varios. Los seres humanos, por ejemplo, poseen 46

cromosomas en total.

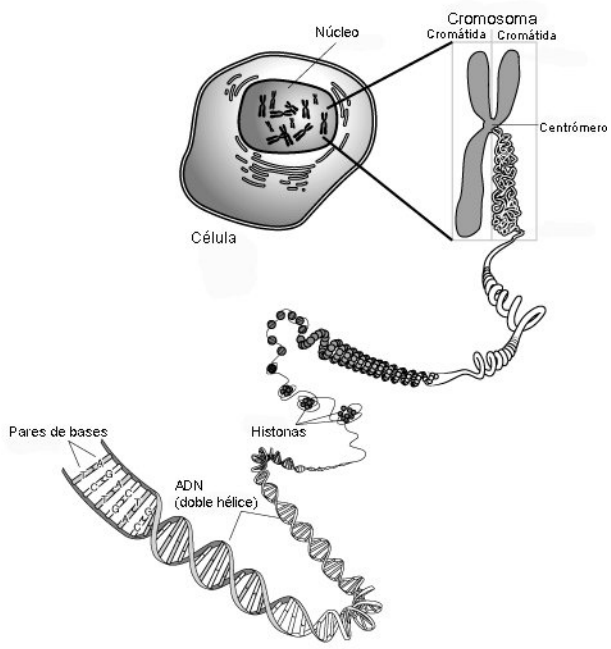

Figura 2.3: *Estructura del cromosoma.*

A lo largo de la hebra del ADN, existen secuencias de bases nitrogenadas que contienen información genética necesaria para la producción de componentes celulares como ARNs y proteínas. Estas secuencias se denominan **genes** y son responsables de "indicar" a las células cómo, cuándo y dónde producir todas las estructuras necesarias para la vida. Todas las células de un mismo organismo poseen la misma información genética, es decir, los mismos genes independientemente de la funcionalidad de la célula. No obstante, en cada célula solamente algunos genes estarán activados y esta activación (**expresión génica**) puede ser temporal.

Los genes, como muestra la Figura 2.6.a, pueden estar formados por **exones** e **intrones**, es decir, secuencias codificantes y no codificantes para este gen respectivamente. Los genes

de los procariotas suelen estar constituidos exclusivamente por exones, mientras, los genes de eucariotas, en general, están formados por las dos estructuras mencionadas.

El ADN es capaz de reproducir una copia de sí mismo a través de la **replicación** (Figura 2.4), lo que permite que las células sean renovadas. En este proceso las hebras son separadas base a base, a partir de la parte superior de la doble hélice, por una proteína llamada helicasa, formando una estructura similar a una Y, la horquilla de replicación. Cada hebra del ADN original sirve de plantilla para la nueva hebra que será creada (hebra complementaria). La lectura de cada base se lleva a cabo por la enzima ADN polimerasa, que se mueve a lo largo de la hebra de plantilla y a cada base escribe su complemento en la hebra complementaria.

Las nuevas hebras de ADN pasan por un proceso de corrección, donde los nucleótidos insertados erróneamente son corregidos utilizando las hebras de plantilla. Sin embargo, este mecanismo no es infalible y, a veces, errores en las nuevas hebras no son detectados o reparados. Estos errores se convierten en **mutaciones** y pueden ser responsables tanto de enfermedades como de la evolución de una especie.

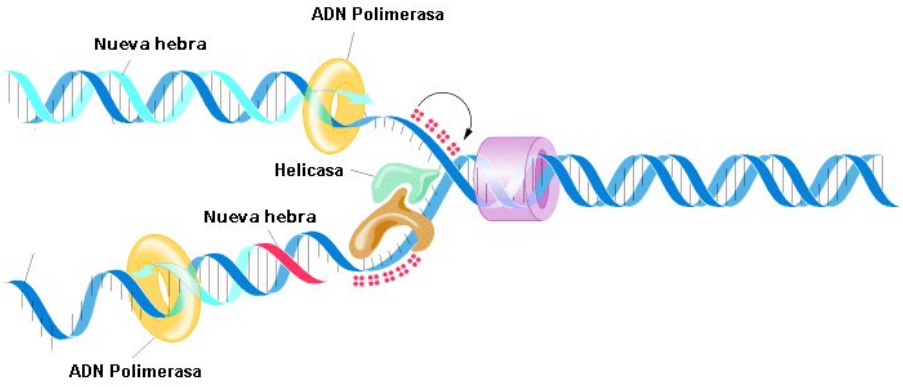

Figura 2.4: *Esquema simplificado de la replicación del ADN.*

Las hebras complementarias producidas en la replicación de ADN son, entonces, utilizadas durante la **transcripción** (Figura 2.5.a) como plantilla para la producción de **ARNm**, o

ARN mensajero. La ARN polimerasa lee los nucleótidos de la hebra de ADN y transcribe la base complementaria en la molécula de ARN. En este proceso, todo el gen es transcrito, tanto sus exones y en caso de que existan, sus intrones (Figura 2.6.b). Como se puede ver en la Figura 2.6.c, el ARNm está formado solamente por exones, puesto que los intrones son removidos durante el proceso de **edición**, también llamado **splicing**.

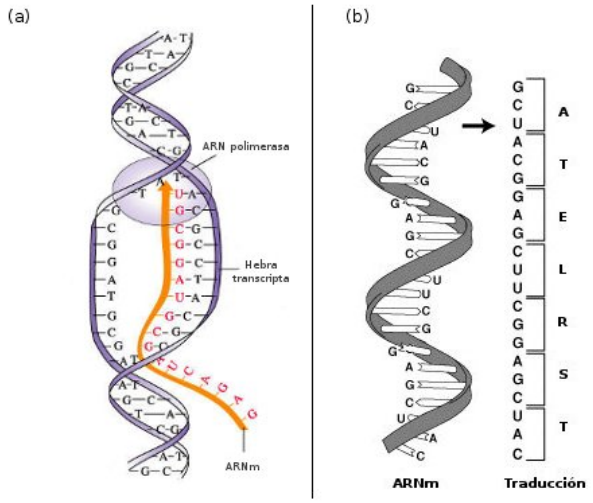

Figura 2.5: *(a) Esquema simplificado de la transcripción. (b) Esquema simplificado del empleo del código genético en la traducción.*

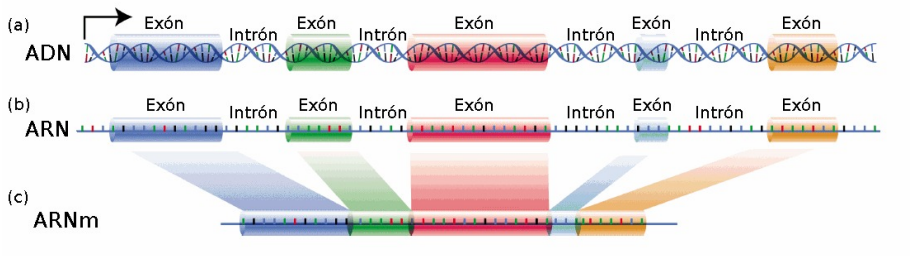

Figura 2.6: *Esquema de la construcción del ARNm.*

El *splicing* solamente ocurre en eucariotas, puesto que el genoma de los procariotas no posee intrones, y tiene el objetivo de preparar el ARNm para la traducción. Cuando muchos intrones están presentes en el tránscrito no editado, el proceso de edición puede resultar

en varias combinaciones de exones (Figura 2.7). El hecho de generar diferentes ediciones a partir del mismo tránscrito se denomina **splicing alternativo** y permite que los genes sean expresados de diferentes formas y que, consecuentemente, sean capaces de codificar más de una proteína.

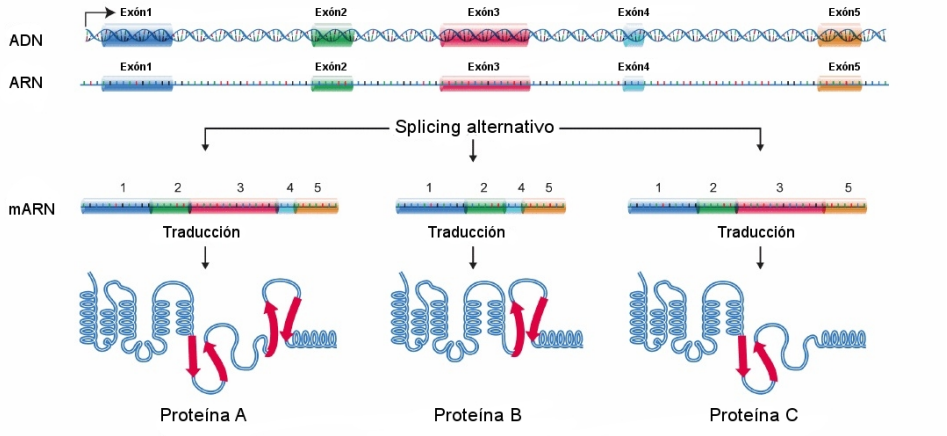

Figura 2.7: *Ejemplo de splicing alternativo.*

A continuación, la secuencia contenida en el ARNm es leída por el ribosoma de triplete en triplete de nucleótidos. Cada triplete de nucleótidos es denominado **codón** y está encargado de codificar un **aminoácido**. Mediante el código genético (Figura 2.8), cada codón es **traducido** a un aminoácido. En la Figura 2.5.b, por ejemplo, el codón GCU es traducido a alanina (A), ACG a treonina (T), GCG a ácido glutámico (E) y así sucesivamente.

Los aminoácidos son estructuras formadas por un grupo amina, un grupo carboxilo, un átomo de carbono y una cadena lateral que designa el aminoácido entre un grupo de veinte posibilidades (Figura 2.9).

Las **proteínas** se forman por la unión de varios aminoácidos por medio de enlaces peptídicos. Son responsables de todos los procesos metabólicos producidos en el organismo y están presentes en todas sus células[40].

Primera base		Segunda base U		Segunda base C		Segunda base A		Segunda base G		Tercera base
U		UUU	Phe [F]	UCU	Ser [S]	UAU	Tyr [Y]	UGU	Cys [C]	U
		UUC	Phe [F]	UCC	Ser [S]	UAC	Tyr [Y]	UGC	Cys [C]	C
		UUA	Leu [L]	UCA	Ser [S]	UAA	Ter [end]	UGA	Ter [end]	A
		UUG	Leu [L]	UCG	Ser [S]	UAG	Ter [end]	UGG	Trp [W]	G
C		CUU	Leu [L]	CCU	Pro [P]	CAU	His [H]	CGU	Arg [R]	U
		CUC	Leu [L]	CCC	Pro [P]	CAC	His [H]	CGC	Arg [R]	C
		CUA	Leu [L]	CCA	Pro [P]	CAA	Gln [Q]	CGA	Arg [R]	A
		CUG	Leu [L]	CCG	Pro [P]	CAG	Gln [Q]	CGG	Arg [R]	G
A		AUU	Ile [I]	ACU	Thr [T]	AAU	Asn [N]	AGU	Ser [S]	U
		AUC	Ile [I]	ACC	Thr [T]	AAC	Asn [N]	AGC	Ser [S]	C
		AUA	Ile [I]	ACA	Thr [T]	AAA	Lys [K]	AGA	Arg [R]	A
		AUG	Met [M]	ACG	Thr [T]	AAG	Lys [K]	AGG	Arg [R]	G
G		GUU	Val [V]	GCU	Ala [A]	GAU	Asp [D]	GGU	Gly [G]	U
		GUC	Val [V]	GCC	Ala [A]	GAC	Asp [D]	GGC	Gly [G]	C
		GUA	Val [V]	GCA	Ala [A]	GAA	Glu [E]	GGA	Gly [G]	A
		GUG	Val [V]	GCG	Ala [A]	GAG	Glu [E]	GGG	Gly [G]	G

Figura 2.8: *El código genético.*

Las moléculas de proteínas pueden escribirse como combinaciones de las letras de los veinte aminoácidos y poseen tres niveles estructurales. La **estructura primaria** es su organización más básica y consiste en una secuencia de aminoácidos (Figura 2.10.b). A veces, puede ocurrir el plegamiento entre aminoácidos de la estructura primaria formando giros, hélices o láminas, estas estructuras bidimensionales constituyen la **estructura secundaria** de la proteína. El conjunto de estructuras secundarias plegadas en el espacio forma la **estructura terciaria**. La **predicción de la estructura de proteínas** es un proceso complejo y laborioso cuyo objetivo es inferir la estructura terciara de una proteína con base en su estructura primaria.

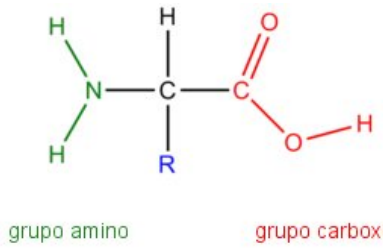

Figura 2.9: *Composición química del aminoácido, donde R es la cada lateral.*

Las proteínas poseen partes que pueden evolucionar, funcionar y existir independientemente de su cadena. Estas regiones son denominadas **dominio proteico** y suelen ser zonas de elevada densidad debido a la acumulación de plegamientos. Una proteína puede tener uno o más dominios y éstos pueden ser funcionales, si son una unidad modular de la proteína que lleva a cabo una función bioquímica determinada, o estructurales, si se refieren a un componente estable de la estructura.

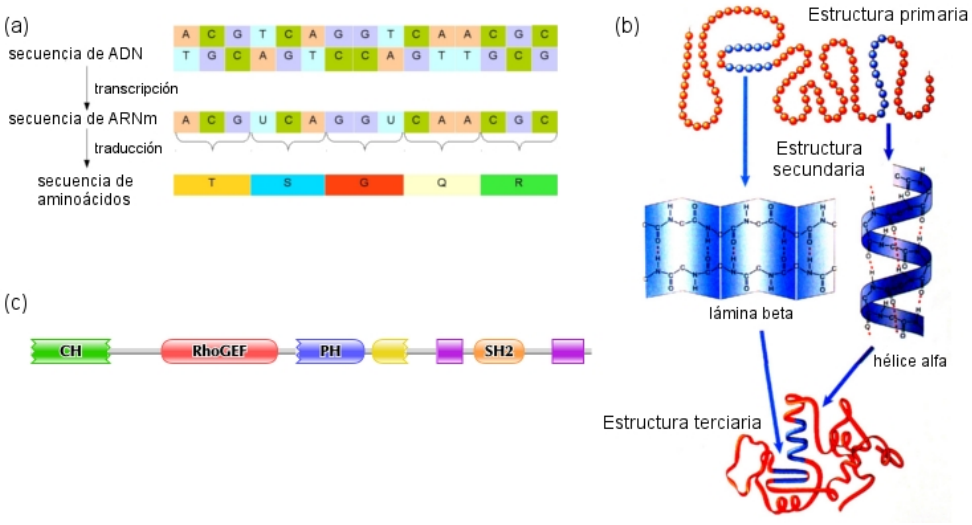

Figura 2.10: *(a) Simplificación del dogma central de la Biología. (b) Niveles estructurales de las proteínas. (c) Ejemplo de proteína formada por diferentes dominios.*

Las proteínas con una similitud significativa en la estructura primaria y/o con estructuras terciarias y funciones similares se dice que pertenecen a la misma **familia de proteínas**.

Existe una fuerte relación evolutiva dentro de cada familia proteica. En general, las proteínas que comparten dominio son agrupadas en una familia de proteínas.

Los miembros de una familia de proteínas se denominan **proteínas homólogas** u **homólogos** (Figura 2.11). Si dos proteínas homólogas están presentes en la misma especie, éstas se conocen como **parálogos**; por otro lado, si están en diferentes especies son **ortólogos**. El proceso de seguir el rastro de la evolución consiste inicialmente en identificar familias apropiadas de proteínas homólogas y, a continuación, utilizarlas para reconstruir trayectorias evolutivas[29].

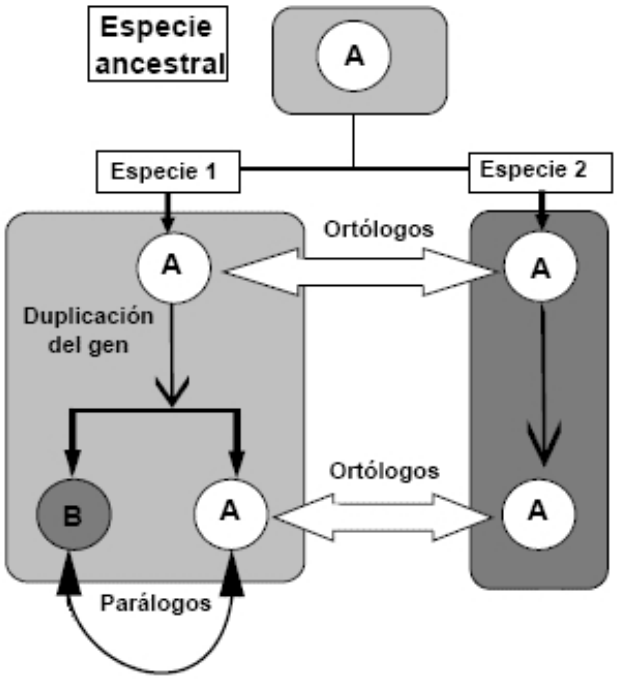

Figura 2.11: *Relaciones de homología entre las proteínas.*

Los seres vivos tienen proteínas, denominadas **enzimas**, que actúan como catalizadores para las reacciones químicas del organismo. Existe un gran número de enzimas y todas las

transformaciones que ocurren en las células las utilizan.

Para que ocurra la catálisis enzimática de una transformación o reacción química, es necesario que el compuesto que va a ser transformado se una a la enzima, sin embargo, casi todas las enzimas son mucho más grandes que los sustratos sobre los cuales actúan, por lo tanto, solamente una pequeña región de la enzima está involucrada en esta reacción. Esta zona específica, marcada en color rojo en la Figura 2.12, es conocida como **centro activo**, resulta siempre bastante conservada en términos evolutivos y es esencial para el funcionamiento correcto de las células. Cualquier alteración en el centro activo imposibilita su efecto catalítico, produciendo así un estado patológico. La localización del centro activo de una enzima es un paso crucial tanto para la investigación fundamental como para el diseño de drogas[41].

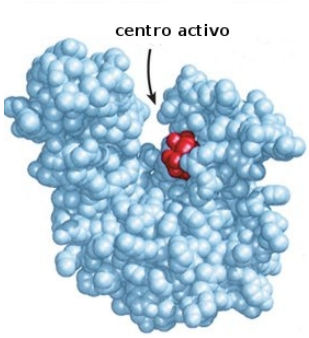

Figura 2.12: *Estructura terciaria de una proteína y su centro activo.*

Genoma es el conjunto completo de información genética contenida en el cromosoma. Tanto el genoma como las secuencias de ADN pertenecientes a él se miden contando su número de **pares de bases** (bp). Para secuencias muy largas, como de un genoma completo, se utiliza kbp, Mbp y Gbp. La secuencia de nucleótidos de la Figura 2.10.b, por ejemplo, contiene 21bp.

De acuerdo con la especie, el genoma puede estar compuesto desde centenas de miles a billones de pares de bases. No obstante, la longitud del genoma no está relacionada con el grado de complejidad de una especie, de hecho, organismos similares pueden diferir en el tamaño de sus genomas y organismos que parecen más complejos pueden poseer genomas más reducidos que otros más simples.

El genoma humano, por ejemplo, posee 3Gbp donde existen de 25 a 30 mil genes, mientras que el genoma del saltamontes posee 180GBbp. El número de genes en un genoma tampoco tiene relación con el tamaño del genoma. Por otro lado, el número de cromosomas puede influir en el número de pares de bases de una especie dada.

El tamaño del genoma de algunos organismos, también, puede explicarse por la presencia de **repeticiones** de secuencias de ADN[36]. Estas secuencias varían de 150 a 300 bp y se repiten miles de veces. A pesar de que no codifican proteínas, la evidencia sugiere que estas repeticiones controlan la transcripción. Existen otras secuencias no codificantes dentro del ADN, como las **zonas de regulación**, responsables de alterar la expresión génica; los **pseudogenes**, genes disfuncionales que han perdido su habilidad de codificar proteínas o que no se expresan más en la célula; y los **transposones**, capaces de copiarse e insertarse en otras partes del genoma; tratar de definir todas estas secuencias desborda el ámbito de este trabajo.

El conjunto de todas las proteínas codificadas por un genoma se denomina **proteoma**. Por otro lado, la colección integral de ARNm forma el **transcriptoma** de una célula o de un organismo.

La **genómica** es el área de la Biología que se dedica a estudiar el genoma, mientras que la **transcriptómica** se enfoca en el transcriptoma y la **proteómica** en el proteoma. Estos tres campos conjuntamente con algunos otros pertenecen a la **Bioinformática**. La

Bioinformática comprende la investigación, el desarrollo y la aplicación de herramientas computacionales y de cualquier otro tipo de abordaje que permita la expansión del uso de los datos biológicos, médicos y relacionados con la salud, incluyendo la adquisición, almacenaje, organización, análisis o visualización de éstos[37].

2.2. El análisis de secuencias

Uno de los problemas iniciales enfrentados por la genómica es cómo medir la similitud de secuencias de ADN y proteínas, tanto dentro de un mismo genoma, como dentro de genomas diferentes de especies iguales o dentro de genomas de especies diferentes.

Se sabe que el ADN y las proteínas pueden ser similares en relación a su función, estructura o secuencia de nucleótidos o aminoácidos. El alto grado de similitud entre dos secuencias suele indicar que ambas han evolucionado a partir del mismo ancestral (homólogas) y poseen la misma estructura tridimensional[13]. La suposición fundamental para el ADN es que dos secuencias de ADN que son similares probablemente comparten la misma función, aunque ocurran en diferentes partes del genoma o entre dos o más genomas[7]. Por otro lado, la hipótesis fundamental para las proteínas es que la secuencia lineal determina su forma, que a su vez establece su función.

Comparar secuencias, por lo tanto, es esencial para obtener información biológica relevante y fundamental a la hora de asignar características a una nueva secuencia. Esta comparación puede seguir un enfoque por pares o por bloques de secuencias (**alineamiento múltiplo**). Además, las secuencias pueden ser alineadas a través de técnicas de **alineamiento local**, que busca alineamientos entre partes de dos o más secuencias, o de **alineamiento global**, que busca alinear integralmente la secuencia. Cada enfoque y técnica de alineamiento depende de la meta a ser alcanzada por el análisis de secuencias.

Al comparar dos secuencias de ADN cuyas longitudes son superiores a 100bp, se puede decir que éstas son similares si más del 70 % de los nucleótidos son similares. En el caso de comparación de secuencias de aminoácidos, se dice que las secuencias son similares cuando más del 25 % de sus aminoácidos lo son.

2.3. De la secuencia a la anotación

La **secuenciación** es el método por el cual se determina la secuencia de nucleótidos de una muestra de ADN. El tipo de dato para ser secuenciado varía de acuerdo con el objetivo del proyecto y su presupuesto, pudiendo tratarse de células específicas de un organismo, su genoma completo o inclusive el genoma de ecosistemas enteros.

Existen diferentes técnicas de secuenciación tales como Sanger, 454, Solexa, SOLiD, entre otras. Cada método tiene sus ventajas e inconvenientes, especialmente con respecto al costo y longitud de las secuencias obtenidas.

El método de Sanger

El método por dideoxinucleótido o Sanger (Figura 2.14.a) es la técnica de secuenciación de ADN más común[34]. Este método se basa en la replicación del ADN y en el hecho de que los cuatro nucleótidos trifosfatados (dATP, dCTP, cGTP, cTTP), llamados dNTP (Figura 2.13.a), son los compuestos básicos para construir bloques de nucleótidos y que el grupo hidroxilo (OH) del dNTP reacciona con el fosfato de un nuevo dNTP creando un puente de hidrógeno que permite la incorporación de la nueva base a la cadena.

El método de Sanger emplea dideoxinucleótidos, ddNTP (Figura 2.13.b), que carecen de uno de los OH de manera que cuando uno de estos nucleótidos se incorpora a una cadena de DNA en crecimiento, esta cadena no va a poder continuar su elongación.

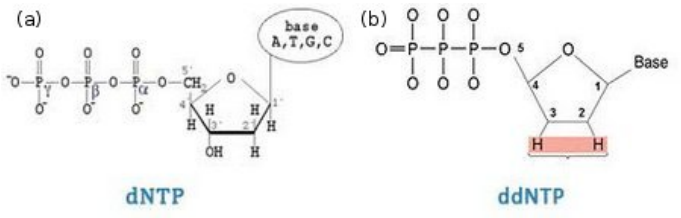

Figura 2.13: *(a) dNTP. (b) ddNTP.*

Inicialmente el fragmento de ADN genómico es aislado, clonado y desnaturalizado para obtener una única hebra de ADN. A la hebra obtenida se le añade un cebador o *primer* (una secuencia de 20 nucleótidos complementaria a la secuencia a ser secuenciada) que se encarga de suministrar el OH que necesita la ADN polimerasa. En presencia de los dNTP y de la polimerasa se sintetiza una secuencia. Esta síntesis puede inhibirse añadiendo un ddNTP, impidiendo así que la secuencia se extienda.

Se preparan cuatro tubos de reacción conteniendo la hebra del ADN para ser secuenciada, la ADN polimerasa, el cebador y los cuatro dNTP. A cada tubo se añade uno de los cuatro ddNTP y a continuación, se producen cadenas de ADN con diferentes longitudes, terminando todas con el ddNTP añadido al tubo.

El resultado de la secuenciación, es entonces, procesado por un aparato de **electroforesis capilar**, que ordena de forma creciente los fragmentos de ADN de acuerdo con su longitud. El primer fragmento más corto será la primera base, el segundo la segunda y así sucesivamente. La salida de la electroforesis capilar es un gráfico llamado **electroferograma** (Figura 2.14.b), que contiene cuatro canales, una para cada nucleótido. La información de este gráfico es utilizada para el procedimiento llamado ***basecalling***, que consiste en llamar, uno a uno, los datos de cada base secuenciada y generar la secuencia.

Es importante tener en cuenta que la salida de la secuenciadora son varios fragmentos de ADN, más pequeños que el original, y que éstos son procesados a través de electroforesis

capilar, generando un electroferograma para cada fragmento.

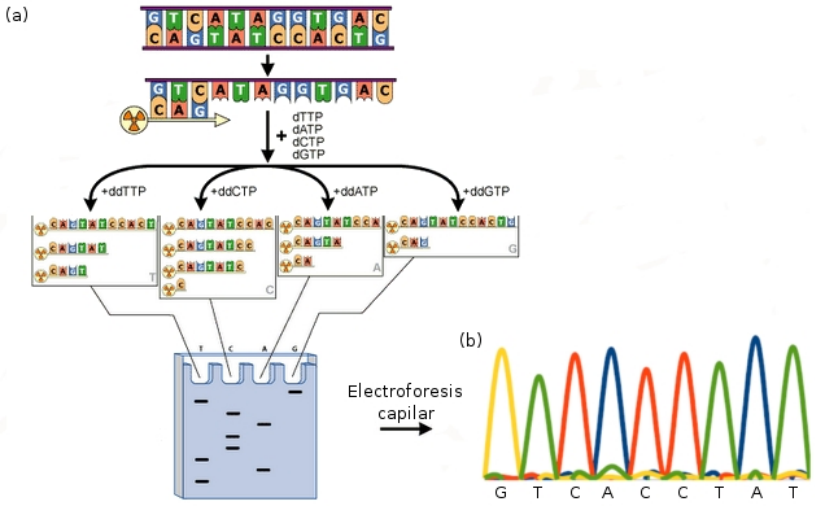

Figura 2.14: *(a) Secuenciación por el método de Sanger. (b) Salida de la electroforesis capilar: el electroferograma.*

El ABI 3730, que utiliza el método Sanger produce, en promedio, 96 secuencias a cada ejecución (*run*), donde las secuencias varían de 800-1100 bp[16]. Sin embargo, en los últimos años, nuevos métodos de secuenciación conocidos como "métodos de nueva generación" describen plataformas que producen gran cantidad (en general millones) de lecturas con pequeñas longitudes (25-400 bp)[16] a bajo costo. Uno de estos avances es precisamente la utilización de la técnica de pirosecuenciación, como el 454 de Roche, que es capaz de generar más de 10^6 secuencias de salida con longitud de 400 bp a cada 10 horas[1].

A las secuencias de salida de la secuenciadora se suele llamar **lecturas** o ***reads***. Cuando se utiliza algún método de secuenciación que obtiene lecturas largas, como el método Sanger, y la entrada de la secuenciadora es un fragmento de ADN complementario (ADNc), decimos que cada lectura de salida es un **EST** (*Expressed Sequence Tag*).

El ensamblaje

El conjunto de lecturas secuenciadas se encuentra desordenado, como las piezas de un puzzle antes de resolverlo. Para obtener el fragmento original de ADN se deben encajar todas las piezas hasta finalizar el puzzle. Este proceso se denomina **ensamblaje de secuencia** y tiene como objetivo determinar el orden de los fragmentos del ADN secuenciado[34].

La salida del ensamblador es un conjunto de ***contigs*** que son formados por el alineamiento de las secuencias de entrada. El número de *contigs* es bastante inferior al número de fragmentos de entrada, pues cada *contig* puede formarse por varios fragmentos. En la Figura 2.15, por ejemplo, el Contig102 está formado por seis ESTs.

Al ensamblar un conjunto de fragmentos es improbable que todos los fragmentos vayan a formar parte de los *contigs* resultantes. Estas secuencias que no forman ningún *contig* se denominan **singletons**.

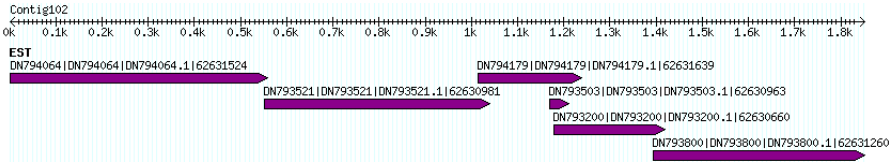

Figura 2.15: *Conjunto de ESTs formando un contig.*

Dependiendo del objetivo del proyecto, el ensamblaje puede detenerse a nivel de *contigs*. No obstante, para proyectos que tienen como objetivo el construir el genoma de un organismo, es imprescindible crear secuencias más largas que un *contig*.

Un **supercontig** es precisamente una secuencia aún más larga que un *contig*, formada por la unión de dos o más *contigs*. Una manera de generar *supercontigs* consiste en disminuir la longitud mínima requerida para solapar dos secuencias, de esta manera, las secuencias que no se unieron porque el número de bases que solapaban era inferior a la longitud inicial, podrán unirse formando estructuras con más pares de bases. Es importante tener en cuenta

que la aplicación de este método depende de los datos y de estudios previos. La unión de dos o más *supercontigs* forma un **scaffold**, resultando que varios *scaffolds* van a crear finalmente la estructura del cromosoma.

Actualmente, existen varios programas de ensamblaje que utilizan diferentes algoritmos y que están disponibles bajo diversos tipos de licencia, CAP3[26], PHRAP[23] y MIRA[12], por ejemplo, pueden descargarse por medio de licencia académica, mientras Newbler[2] es software propiedad de Roche.

Anotación

Después del ensamblaje, las secuencias pasan a la fase de **anotación** (Figura 2.16), donde se extrae la información "oculta" en la secuencia con el objetivo de identificar genes. En esta fase se añade información pertinente sobre los genes existentes, las proteínas que codifican o comentarios a cada secuencia. Existen dos campos de la genómica dedicados a esta tarea: anotación estructural y funcional.

La **anotación estructural** es responsable de detectar genes, sus localizaciones en la secuencia, la estructura de los intrones y exones que lo componen y predecir las secuencias de proteínas que codifican. Por tanto, se pueden utilizar diferentes métodos:

- Método *ab initio*: procedimiento que utiliza solamente las propiedades de la secuencia de ADN para predecir la localización de genes[16]. Este método se basa en sensores y detectores de contenido para discriminar las regiones codificantes y no codificantes y, a partir de ellas, inferir dónde está el gen.
- Método basado en homología: utiliza conceptos de la conservación evolutiva, como la homología, para deducir la localización y estructura de genes.
- Método híbrido: mezcla los dos métodos anteriores.

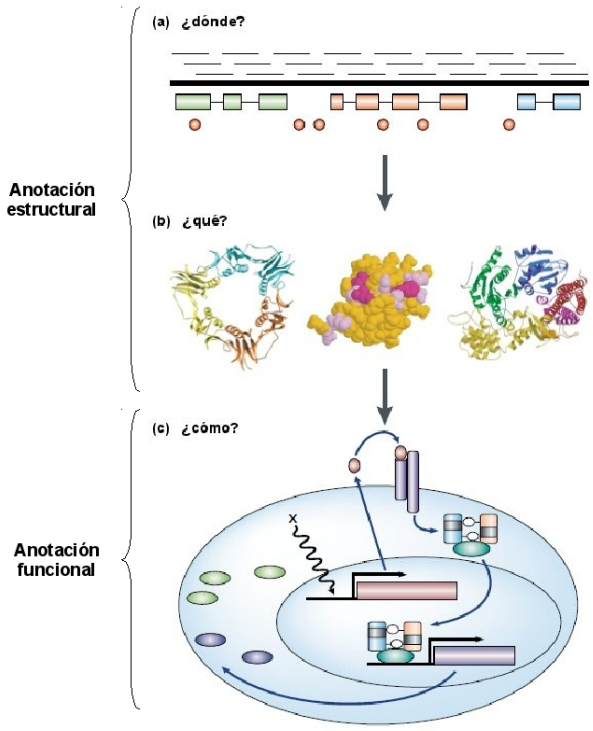

Figura 2.16: *El proceso de anotación se divide en anotación estructural, que a su vez puede ser a nivel de nucleótidos (a) o de proteínas (b), y anotación funcional (c).*

La anotación estructural puede realizarse a nivel de nucleótidos y a nivel de proteínas, sin excluirse mutuamente. La anotación a nivel de nucleótidos pregunta "¿dónde?" están las partes relevantes del genoma y tiene como objetivo identificar puntos de referencia genómica, genes, ARN no codificantes, regiones de regulación, repeticiones, duplicaciones y mutaciones. Por otro lado, la anotación a nivel proteico contesta a la pregunta "¿qué?", buscando recopilar un catálogo definitivo de las proteínas de los organismos a la vez que las nombra[45].

Existe una infinidad de programas que pueden ser utilizados para diferentes objetivos de la anotación estructural. **BLASTn**[4] y programas basados en búsqueda de similitudes buscan puntos de referencia genómica, mientras **BLASTx**[4] puede ser utilizado para encontrar genes y **RepeatMasker**[42] es aplicado para identificar y mapear repeticiones.

La **anotación funcional** tiene como meta responder a la pregunta "¿cómo?", prediciendo la función biológica de los genes y proteínas y su proceso biológico. Con tal propósito, se utiliza un vocabulario estándar llamado **Gene Ontology**[46] (GO).

GO es una ontología que consta de tres partes: función molecular, proceso biológico y componente celular. Está organizada como una jerarquía de términos, lo que permite diferentes niveles de especificidad en la anotación, de acuerdo con la exactitud de su predicción. Por ejemplo, el término "enzima" puede conducir a términos específicos como "liasa", "carbono-oxígeno liasa" ,"hidrolasa" y "treonina deshidratasa".

Una de las maneras de obtener los términos GO de una secuencia dada es utilizando el programa **Blast2GO**[14].

2.4. Algunas Bases de Datos bioinformáticas

La investigación bioinformática utiliza diferentes tipos de Base de Datos (BD), de acuerdo con su campo de acción. Actualmente, están disponibles repositorios con información de secuencias de ADN, de proteínas o de dominios de proteínas, entre otros. Estos repositorios son actualizados periódicamente y almacenan la información en formatos propios, o sea, no existe un patrón con un formato determinado.

Base de Datos de secuencias de nucleótidos

Las BD de ADN constituyen el núcleo de la genómica y el fundamento de la investigación bioinformática[16]. Existen varias BD de ADN, creadas para diferentes propósitos y a veces con diferentes anotaciones. El mayor repositorio de secuencias de ADN es el INSDC (International Nucleotide Sequence Database Collaboration), formado por el DDBJ[11] (DNA Data Bank of Japan), por el GenBank[8] y por el EMBL[28] (European Molecular Biology Laboratory). Estas tres entidades colaboran entre sí e intercambian información diariamente.

GenBank

El GenBank es una extensa BD pública, mantenida por el NCBI (National Center of Biotechnology Information), que contiene una colección anotada de todas las secuencias de ADN disponibles públicamente. Actualmente, se encuentran 124,277,818,310 bp, en un total de 132,015,054 entradas, en su principal división[31].

EMBL

EMBL es una BD de secuencia de nucleótidos, mantenida por el EBI (European Bioinformatics Institute), que contiene 301,588,430,608 nucleótidos con 199,575,971 entradas, donde humano, ratón, buey y pez cebra están entre las especies más relevantes[17].

Base de Datos de secuencias de proteínas

El Consorcio UniProt[47] ha realizado esfuerzos significativos para integrar la información asociada a secuencias de proteínas. Esta información se organiza en cuatro repositorios principales de dominio público, siendo el UniProt Knowledgebase (UniProtKB) la BD central. El UniProtKB es una colección de datos curados manual y automáticamente, provenientes del UniProt/Swiss-Prot[9] y del UniProt/TrEMBL[5] respectivamente.

Base de Datos de dominios

Pfam[19] es una colección de familias de proteínas compuesta por Pfam-A, datos de alta calidad que comprenden familias manualmente curadas, y Pfam-B, constituido por entradas generadas automáticamente. A pesar de la baja calidad de los datos de Pfam-B, éstos pueden ser útiles en la identificación de regiones funcionales conservadas cuando ninguna entrada de Pfam se encuentra. Pfam también contiene perfiles basados en alineamientos múltiples y

modelos ocultos de Markov para todos los dominios de proteínas[43], lo que permite generar anotaciones de secuencias más significativas.

2.5. Algunos programas bioinformáticos y sus formatos

El campo de la Bioinformática, en su más pura esencia, se trata de realizar continuas comparaciones[34]. Se comparan secuencias de ADN o ARN con otras secuencias o con BD de secuencias, se comparan la expresión del mismo gen en células tratadas con fármacos con otras sin tratamiento, o se comparan las estructuras tridimensionales de las proteínas y sus respectivas estructuras primarias, sólo por citar algunas posibles comparaciones.

Para poder ejercer las comparaciones que componen el núcleo de la Bioinformática y obtener resultados relevantes es necesario conocer los varios programas disponibles y sus funciones, así como comprender sus formatos. Es cierto que, en este campo, existe una gran variedad de programas y que muchos adoptan diferentes formatos de entrada y salida.

Los formatos más utilizados, en general, son textos planos, relativamente sencillos y fáciles de analizar, entendiendo, por tal, lo que en la Comunidad Bioinformática se expresa con el término *parsear*, al que se referirá de aquí en adelante. Algunos lenguajes de programación, como Perl y Java, suelen tener un conjunto de librerías asociadas (BioPerl[44] y BioJava[25], respectivamente) capaces de extraer información de estos formatos, además de poseer otras aplicaciones bioinformáticas.

A continuación, se comentan algunos programas que se juzgan relevantes para el entendimiento de este trabajo, junto con sus respectivos formatos.

FASTA

FASTA[33] es el nombre de un popular programa de alineamiento de secuencias de nucleótidos y proteínas. Las secuencias de entrada deben seguir el formato mostrado como ejemplo

Tipo de BLAST	Entrada	Base de datos
BLASTn	ADN	Nucleótido
BLASTp	Proteína	Proteína
BLASTx	ADN (traducido a proteína)	Proteína
tBLASTn	Proteína	ADN (traducido a proteína)
tBLASTx	ADN (traducido a proteína)	ADN (traducido a proteína)

Cuadro 2.1: *Los diferentes tipos de BLAST.*

en la Sección A.1, donde la primera linea (denominada linea de definición) empieza siempre por un carácter "$>$" seguido por un identificador único de la secuencia y, opcionalmente, por una breve descripción. Las líneas siguientes contienen la secuencia de ADN o proteína hasta que exista una otra linea de definición o finalice el fichero.

El formato de entrada de FASTA resulta bastante popular porque es simple y fácil de *parsear*. Actualmente, las entradas por defecto de muchos programas bioinformáticos son ficheros FASTA y, por consiguiente, el Sistema Experto que se estructura en este trabajo seguirá la misma tendencia.

BLAST

BLAST[4], abreviatura de *Basic Local Alignment Search Tool*, es una herramienta de comparación de secuencias muy popular y eficiente, que rápidamente es capaz de encontrar, en una BD, secuencias de proteínas o nucleótidos similares a la de entrada. Esta herramienta puede ser utilizada para encontrar genes en un genoma, predecir la función de proteínas o su estructura tridimensional, identificar los homólogos, etc.

BLAST está compuesto por cinco programas que aceptan ficheros FASTA (de ADN o proteínas) como entrada y son capaces de realizar búsquedas en BD (de ADN o proteínas), conforme muestra el Cuadro 2.1.

El fichero de salida de BLAST (ver ejemplo en la Sección A.2) tiene algunos elementos importantes, especialmente, para la anotación de secuencias:

- ***query:*** es la secuencia de entrada, identificada por un nombre único en el fichero FASTA.
- ***hit:*** es el conjunto de secuencias similares a la *query* encontradas en la BD empleada para los alineamientos. Este conjunto puede tener cero o varios *hits*, en el primero caso aparece la salida "No hits found".
- ***e-value*:** la estimación del número de veces que se podría esperar el resultado obtenido por casualidad. El valor del *e-value* siempre es mayor o igual a cero. Cuanto menor sea el *e-value*, más similares son las secuencias y más confianza se puede tener que el *hit* es realmente homólogo a la *query*. Pero solamente el hecho de que un *e-value* sea menor que otro no garantiza que la secuencia de menor *e-value* sea la asignación correcta.
- **porcentaje de identidad:** número de residuos (base de nucleótido o aminoácidos) idénticos dividido por el número de residuos alineados. Este valor no es un sustituto para el *e-value*, puesto que no tiene en cuenta la longitud del alineamiento. Por ejemplo, una identidad de 100 % entre una *query* de 864bp y *hits* con cerca de 20bp, como en la Figura 2.17, puede o no ser relevante.
- **longitud del alineamiento:** indica la longitud del alineamiento entre la *query* y el *hit*. Es una variable que se debe tener en cuenta a la hora de analizar si el *hit* obtenido tiene o no algún significado.
- ***bit score*:** indica cuan bueno es el alineamiento. Cuanto mayor el *bit score*, mejor el alineamiento.

BLAST2GO

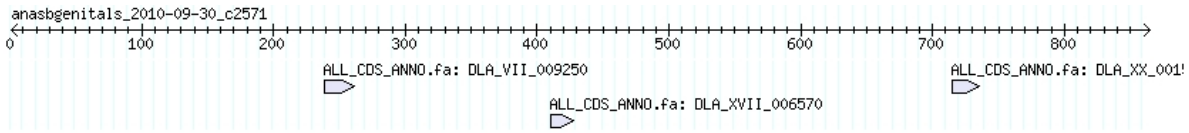

Figura 2.17: *Alineamiento de una secuencia de 864bp con sus tres hits, cada uno con cerca de 20bp y 100 % de identidad.*

BLAST2GO[14] es una herramienta de anotación funcional capaz de asignar términos GO a secuencias de ADN a través de técnicas de minería de datos. Este programa tiene como entrada un fichero en formato FASTA y utiliza BLAST contra la BD no redundante de NCBI (*nr*) para encontrar los homólogos correspondientes a las secuencias existentes en este fichero. A continuación, con base en la información generada por BLAST, se obtienen los términos GO relativos a cada secuencia.

La salida de BLAST2GO consiste en un fichero tabular compuesto por tres columnas (Sección A.3), donde la primera contiene el identificador de la secuencia, la segunda el término GO asociado a la anotación y la tercera la descripción del término GO. Es importante tener en cuenta que una secuencia puede estar asociada a diferentes GO debido a su estructura jerárquica.

Pfam scan

Pfam scan[19 24] es un *script* que busca dominios en la librería de modelos ocultos de Markov de Pfam para una secuencia dada. Tiene como entrada un FASTA de proteína y su salida, ejemplo en A.4, es un fichero ASCII que contiene el identificador de la secuencia, las coordenadas de alineamiento de la *query* y del modelo oculto de Markov, el *e-value* relativo al alineamiento, los residuos del centro activo predicho, cuando existan, entre otras informaciones.

Como la entrada del *Pfam scan* es una secuencia de proteína y la secuencia original, a ser anotada, siempre está compuesta por nucleótidos, es necesario convertir los nucleótidos en aminoácidos para poder ejecutar el *script*. Esta conversión, de hecho, es la traducción, explicada anteriormente en 2.1, aplicada tanto a secuencias de ARN como de ADN.

Es importante tener en cuenta que al traducir una secuencia de nucleótidos a proteína, esta traducción puede ser hecha utilizando tres posiciones de inicio diferentes, denominadas **marcos de lectura** o ***frames***. Por ejemplo, la secuencia ATCGATGTACGC, si se lee a partir del primer *frame* contiene los codones: ATC, GAT, GTA y CGC. En caso de que el punto de partida sea el segundo *frame*, los codones serán: TCG, ATG, TAC; y, en el último caso: CGA, TGT, ACG. Cubrir todos los *frames* es importante, porque, como se ha explicado previamente en la Sección 2.1, diferentes aminoácidos están asociados a distintos codones.

Además, para sacar toda la información posible de la secuencia a analizar, se debe traducir la secuencia original y también su complemento, siempre teniendo en cuenta los diferentes *frames*.

La salida de la traducción de una secuencia, será, por lo tanto, seis secuencias diferentes, agrupadas de acuerdo con la orientación de la hebra (**strand**).

Capítulo 3

Consideraciones relevantes en la anotación de genes

3.1. El problema de la anotación de genes

El valor de un genoma depende de la exactitud de su anotación, no obstante, detectar genes, su organización, estructura y función entraña un gran desafío en la era genómica y pos-genómica[22]. La anotación puede realizarse automática o manualmente, donde cada enfoque tiene sus ventajas e inconvenientes.

Es evidente, que la interpretación humana del análisis crudo a través de la **anotación manual** genera datos con mayor calidad y estructuras de genes más exactas[35]. Pero este proceso es bastante lento y requiere expertos con conocimientos bioquímicos y familiarizados con los diversos programas aplicados en los diferentes pasos.

La **anotación automática**, por otro lado, es más rápida, no requiere un equipo cualificado de anotadores y procesa los datos de forma consistente. Sin embargo, puede predecir menos genes de los que realmente existen, sus resultados son más generales, no son tan exactos y son más propensos a errores. De hecho, para conseguir anotaciones automáticas fiables, éstas deberían ser revisadas manualmente por un experto.

Con el advenimiento de nuevas técnicas capaces de secuenciar genomas completos a bajo coste y en poco tiempo, se está secuenciando una gran cantidad de organismos y, consecuentemente, generando un gran número de secuencias que deben ser ensambladas y anotadas.

Para evitar el cuello de botella de la curación manual, la anotación de gran cantidad de genes debe ser automática. Por lo tanto, gran parte de las anotaciones disponibles en los repositorios genómicos es fruto de dicha automatización. Para beneficiarse del poder de la secuenciación genómica, las herramientas de anotación deben ser fiables y las Bases de Datos (BD) consistentes[6]. No obstante, la anotación automática puede producir datos de baja calidad e incluso, anotaciones erróneas.

Como las anotaciones realizadas por diferentes grupos de trabajo se realizan automáticamente, y son añadidas a las BD públicas sin una curación previa y empleadas en la anotación de otros datos, la existencia de anotaciones incorrectas conduce a errores en la anotación de otras secuencias, que también serán depositadas en estos repositorios y contribuirán a la acumulación de futuros errores.

La anotación manual puede mejorar la anotación automática, puesto que estas anotaciones van a enriquecer positivamente las BD que serán utilizadas posteriormente para anotar nuevas secuencias. Esta contribución es especialmente importante para la anotación de genomas de especies cercanas a la especie anotada manualmente.

3.2. *Pipelines* de anotación

Uno de los desarrollos claves de los últimos años ha sido la implementación automática de *pipelines* de anotación genómica[38] para manipular grandes cantidades de datos de secuencias. Estos *pipelines* unen diferentes algoritmos, y métodos desarrollados en *software*, cada uno con sus propios parámetros y formatos de entrada y salida, donde cada capa del proceso establece un mayor nivel de refinamiento de la anotación.

Desarrollar, mantener y utilizar un *pipeline* de anotación requiere mucho conocimiento bioinformático así como diseños y máquinas de alto rendimiento capaces de ejecutar todo el proceso. Además, algunos pasos clave de los *pipelines* suelen requerir biólogos expertos para tomar importantes decisiones, modificar la BD, comparar resultados intermedios, manejar y convertir ficheros manualmente, lo que implica una intensa labor humana muy propensa a errores[22]. No obstante, los *pipelines* automáticos, generalmente, son una concatenación de programas, donde la salida de uno se convierte en la entrada del siguiente, sin tener en cuenta el razonamiento del experto entre cada paso.

Por otro lado, los *pipelines* semiautomáticos, cuentan con la intervención del experto y, por consiguiente, deben generar resultados más exactos. Sin embargo, la necesidad de la presencia del experto en determinadas etapas de este proceso entraña los mismos inconvenientes de la anotación manual, pues ralentiza el trabajo y requiere personal con entrenamiento y conocimiento para hacerlo.

AGMIAL[10] es un sistema de anotación semiautomática de procariotas desarrollado para permitir a los biólogos de pequeños laboratorios anotar sus genes sin la necesidad de esfuerzos bioinformáticos para instalar o crear *pipelines*. La filosofía central de este sistema consiste en que los expertos constituyen el núcleo del proceso de anotación, donde la función de los ordenadores y de los ordenadores es asistirles en esta tarea compleja. AGMIAL, por lo tanto, requiere la intervención del biólogo a lo largo del proceso de anotación, lo que no soluciona el cuello de botella de la anotación manual.

Una posible solución para esta cuestión es emular el conocimiento del experto en los pasos en que éste debería intervenir en el sistema. Siguiendo este enfoque, la calidad de la anotación aumentaría en relación a un sistema automático y el tiempo de procesamiento se reduciría en relación a un sistema semiautomático o manual.

Ensembl[27] es un proyecto conjunto entre el EMBL-EBI y el Instituto Wellcome Trust Sanger que tiene como objetivo desarrollar sistemas informáticos que producen y mantienen anotaciones automáticas de determinados eucariotas. El *pipeline* de anotación de Ensembl[15,35] es una combinación de programas *software* que utiliza un método de anotación automática basado en un conjunto de reglas, el cual se ha desarrollado observando cómo los expertos en anotación construyen la estructura de los genes. Fue diseñado para anotar el genoma de organismos eucariotas, más específicamente, el genoma humano, y por tanto, tiene un enfoque diferente de un *pipeline* de anotación de procariotas o de secuencias fuera del contexto del genoma.

Este *pipeline*, esquematizado en la Figura 3.1, fue empleado para anotar diferentes especies, como el ratón, la rata, el mosquito, el pez cebra, entre otras; y, de acuerdo con Potter y col.[35], funciona muy bien en su rol actual de análisis de gran cantidad de datos de secuencia. La documentación de instalación y configuración de este *pipeline* está disponible en la web de Ensembl[18]; sin embargo, instalarlo y configurarlo adecuadamente resulta ser una tarea laboriosa, que requiere librerías específicas (algunas obsoletas) y diversos programas como el WU-BLAST[21]. Además, no se sabe bien cuáles son las reglas empleadas para anotar una secuencia dada, lo que hace de este *pipeline* una especie de caja negra sin posibilidades de análisis. Otro inconveniente es la excesiva cantidad de recursos informáticos requeridos para ejecutar los programas en un tiempo computacional razonable, por ejemplo, para alinear todas las proteínas humanas al genoma en algunas horas es necesario 400 CPUs[15].

FIGENIX[22] es una plataforma de anotación automática que utiliza un Sistema Experto (SE) para reemplazar el conocimiento del profesional en varios pasos del proceso de anotación. Este SE modela la pericia del experto y es capaz de emular su conocimiento en determinadas tareas como comparar resultados intermedios de diferentes métodos, modificar la BD y evaluar la relevancia de predicciones.

Esta plataforma contiene ocho *pipelines* diferentes para anotación estructural y funcional, y está disponible gratuitamente para uso académico a través de una interfaz web[22]. No obstante, el empleo de esta herramienta no está difundido en la Comunidad Bioinformática y no se puede probar, puesto que es necesaria una clave de acceso para acceder a su servidor. Además, sus desarrolladores no proporcionan una versión de ejecución local, lo que obliga al usuario a utilizar un servidor ajeno para analizar y almacenar sus datos, que pueden contener información sensible.

3.3. Un Sistema Experto para anotación de secuencias de ADN

Actualmente, existe una gran cantidad de programas bioinformáticos disponibles públicamente, lo que permite crear *pipelines* diferentes que idealmente deberían ser capaces de asignar a la misma secuencia la misma anotación y obtener la misma cantidad de genes. Sin embargo, los servicios de anotación difieren considerablemente en el número de genes, su longitud y/o anotación, como se demuestra en Bakke y col.[6], al comparar tres diferentes sistemas de anotación.

En los últimos años, la preocupación respecto de la veracidad de la información existente en las BD ha aumentado considerablemente, pero todavía no se sabe la cantidad de secuencias que se encuentran anotadas erróneamente. Lo que se conoce es que toda vez que un sistema de anotación produce un error, éste es depositado en las BD, y posteriormente utilizado para confirmar el mismo error, en un caso de reanotación empleando el mismo sistema, o propagar el error en caso de nuevas especies.

La solución para este problema, como se ha visto en la Sección 3.2, puede estar en sistemas expertos de anotación automática, como Ensembl, que basándose en el razonamiento de un especialista en anotación, toman decisiones en las fases críticas del proceso. Pero, hasta ahora, los sistemas con estas características son desarrollados para la anotación de genomas

completos. No obstante, existen varios proyectos dentro de la Comunidad Científica que no trabajan con genomas completos, y sí con porciones del genoma, que necesitan que sus secuencias de ADN sean anotadas. Por consiguiente, desarrollar un SE para anotación de secuencias de ADN sin tener en cuenta el contexto genómico puede contribuir a la mejora de la calidad de los datos que serán futuramente generados y analizados.

De lo expuesto hasta aquí se deducen los siguientes aspectos relevantes:

- En la Comunidad Científica resulta de gran interés el desarrollo de un SE con capacidad de procesamiento automático para anotaciones de secuencias de ADN, fuera del contexto genómico, ampliando así el ámbito de aplicaciones actualmente muy centrado en la anotación de genomas completos.

- Es absolutamente necesario extraer el conocimiento subyacente de los expertos en anotación, para ser aplicado en las diferentes fases del proceso de anotación de forma efectiva.

- La extracción del conocimiento requiere de los mecanismos necesarios para extraer lo relevante del mismo, organizándolo en forma de datos manejables, inferir estructuras de conocimiento válidas, para finalmente inferir las consecuencias, en nuestro caso orientadas a la anotación de secuencias de ADN, que no se encuentran en un contexto genómico.

Evidentemente, se hace necesaria la elección de una metodología capaz de abordar los aspectos anteriores de forma eficiente. CommonKADS resulta ser probablemente la más apropiada para abordar esta problemática dentro de la Ingeniería del Conocimiento, ya que proporciona mecanismos necesarios y apropiados para enfrentar los distintos aspectos anteriormente mencionadas. En efecto, posee estructuras de elicitación del Conocimiento a partir de los expertos, con la suficiente garantía de que el mismo, una vez extraído, resulte de utilidad. Proporciona los mecanismos necesarios para estructurar el conocimiento de

forma eficiente, a la vez que los organiza convenientemente para inferir a partir de ellos los mecanismos propios orientados hacia la anotación automática de secuencias de ADN.

Hasta donde hemos podido investigar, no existe en la literatura un modelo capaz de extraer el conocimiento, organizarlo y estructurarlo para inferir anotaciones de secuencias de ADN, sin tener en cuenta el contexto genómico, de forma automática y eficiente, razón por la cual en este trabajo se propone la aplicación de la metodología CommonKADS como marco estructural fiable. El diseño de un marco general, como el que se propone en este trabajo, constituye una aportación de importancia en el ámbito de la Bioinformática. Su estructura modular y flexible permite además la posibilidad de intercambiar estructuras y procesos sin afectar al diseño general del sistema, lo que le confiere un valor añadido de vital importancia de cara a incorporar en el futuro nuevos procesos o estructuras que puedan aparecer como consecuencia de los avances de la investigación.

En el Capítulo 4, se plantea el diseño del modelo CommonKADS, describiendo sus estructuras y mecanismos orientados a la anotación de secuencias de ADN desde el punto de vista de la anotación estructurada en forma de *pipeline*.

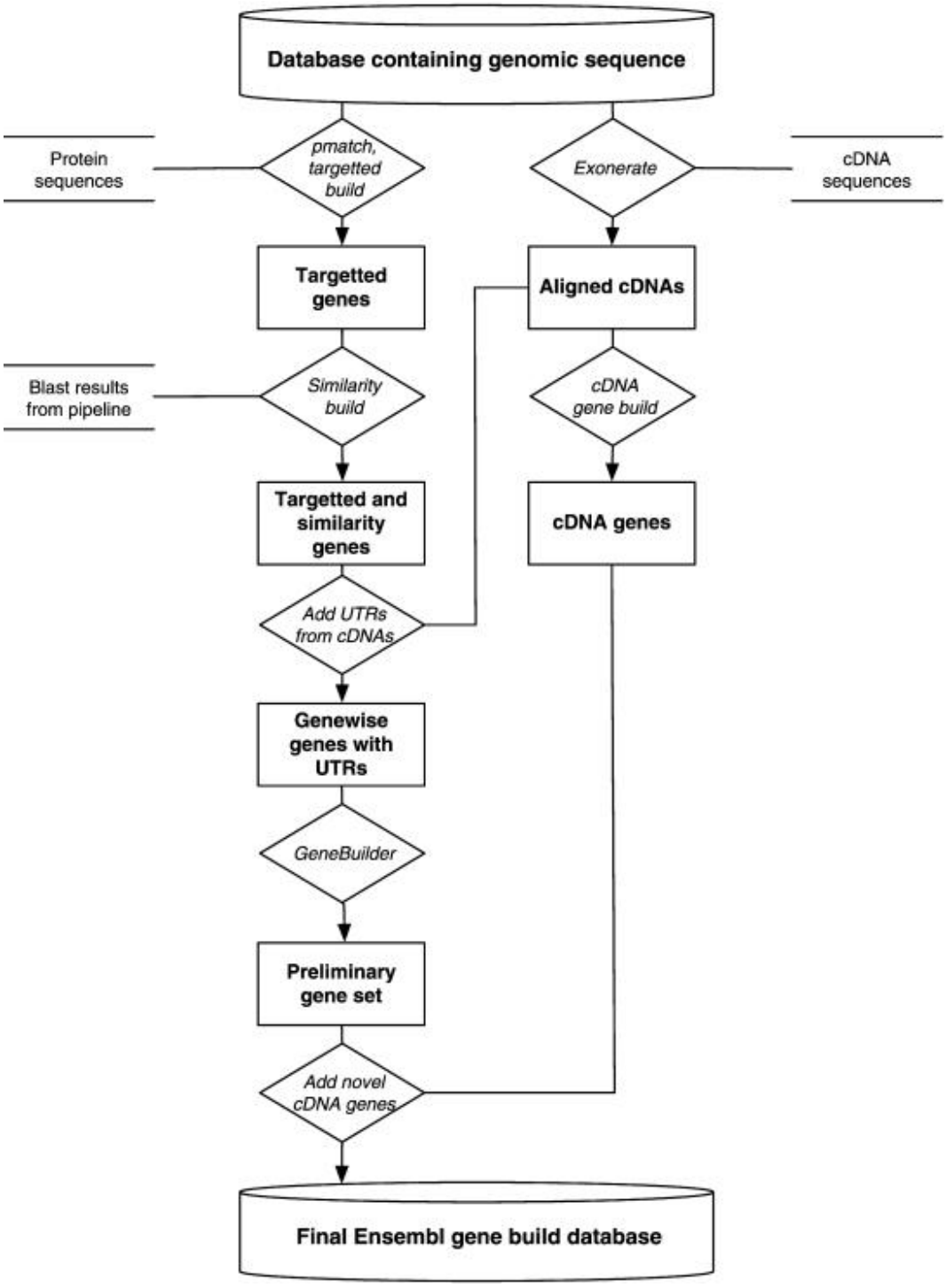

Figura 3.1: *Visión general del pipeline de Ensembl.*

Capítulo 4

Análisis, diseño y realización

4.1. CommonKADS: descripción general

Sistemas Basados en Conocimiento (SBC) son aplicaciones que hacen uso de técnicas basadas en principios de Inteligencia Artificial para solucionar problemas, proporcionando toma de decisiones rápidas y aumento de la productividad y de la calidad de las decisiones. El desarrollo de estos sistemas requiere el entendimiento y la estructuración del conocimiento de tal manera que este pueda ser incorporado a la aplicación.

La **Ingeniería del Conocimiento** tiene como objetivo crear metodologías científicas para el análisis y estructuración del conocimiento, y consecuentemente, para el desarrollo de SBC. Existen diferentes metodologías que soportan la Ingeniería del Conocimiento, entre ellas, se destaca **CommonKADS**.

CommonKADS es una metodología flexible capaz de adaptarse a cualquier problema que ofrece una serie de herramientas para modelar SBC, donde el conocimiento es entendido con base en su contexto y propósito. Su objetivo final consiste en estructurar el proceso de desarrollo propio de la Ingeniería del Conocimiento, que se concreta en un sistema que debe resolver los problemas con una capacidad comparable a la del experto humano como poseedor del conocimiento[32].

Una de las ventajas de esta metodología es su reusabilidad, puesto que después de modelar la información para un determinado problema y dominio, este modelo puede ser aplicado a problemas similares. De esta forma, los esquemas utilizados en este trabjo pueden ser aplicados a problemas que se asemejen a la anotación de secuencias.

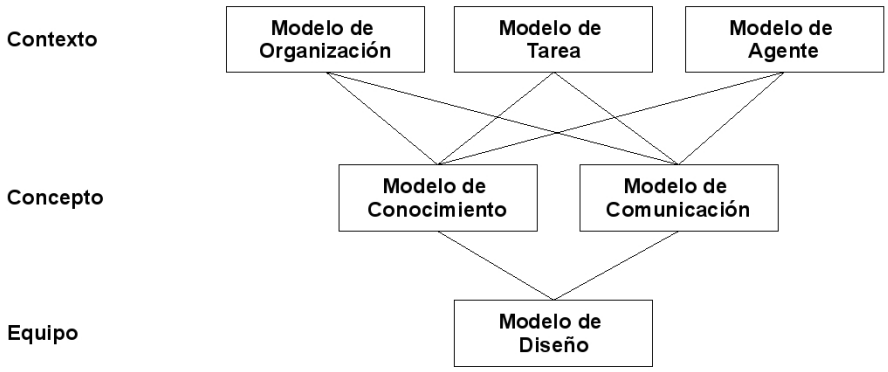

Figura 4.1: *Modelos de la metodología CommonKADS.*

La metodología CommonKADS consiste de un conjunto de seis modelos (Figura 4.1) que especifican todos los aspectos ligados a la aplicación a ser desarrollada, incluyendo la organización, los recursos humanos, los aspectos de implementación y la interacción entre ellos. Estos modelos son agrupados en:

- **Contexto**

  Los modelos relacionados a Contexto responden a los porqués de la implantación de un sistema que involucre conocimiento. Para contestar a las cuestiones relativas al contexto es necesario tener en cuenta tres aspectos clave: la estructura de la organización donde se va a desarrollar el sistema; las tareas que realizará el sistema y su relación con la tareas globales de la organización; y las entidades que estarán involucradas en la

ejecución de estas tareas. Cada uno de estos puntos específicos es estudiado de forma separada utilizando un modelo.

El **Modelo de organización** soporta el análisis de las características principales de una organización, con el objetivo de detectar problemas que puedan ser solucionados a través del empleo de SBC, calcular la viabilidad del desarrollo y ponderar los impactos en la organización de las acciones del conocimiento.

El **Modelo de tarea** analiza la tarea global, sus entradas y salidas, precondiciones y criterios de ejecución, y también los recursos y competencias necesarios para realizarla.

El **Modelo de agente** describe las características de los ejecutores de una tarea (agentes), especificando sus competencias, autoridad para actuar y restricciones, y describe los enlaces de comunicación entre agentes.

- **Concepto**

Los modelos agrupados en esta categoría tienen como enfoque principal la descripción conceptual del conocimiento aplicado en una tarea, teniendo en cuenta el conocimiento necesario para desarrollar el sistema y su transmisión y/o recepción a través de una comunicación.

La finalidad del **Modelo de conocimiento** es explicar los tipos y estructuras de conocimiento utilizados para realizar una tarea, describiendo, de forma comprensible a los humanos, el rol que las diferentes componentes de conocimiento poseen en la solución de problemas. Este modelo es un importante medio para la comunicación con los expertos y los usuarios sobre los aspectos relacionados con la solución de problemas de un sistema de conocimiento tanto en la fase de desarrollo del sistema como a lo

largo de su ejecución. Por otro lado, el **Modelo de comunicación** proporciona la descripción de las transacciones entre los agentes implicados en una determinada tarea.

- **Equipo**

  Esta etapa se focaliza en los aspectos computacionales necesarios para llevar a la práctica toda la información analizada anteriormente e implementar el sistema. Consta únicamente de un modelo, el **Modelo de diseño**, que es responsable de la especificación técnica del sistema necesaria para implementar las funciones propuestas en los modelos de conocimiento y comunicación. Este modelo describe, por ejemplo, la arquitectura, la plataforma de implementación, los módulos de software, etc.

Se sabe que analizar el contexto donde se va a implementar el SBC es importante para evitar su fracaso y potenciar su uso.

En el caso que nos ocupa, la organización a beneficiarse del Sistema Experto (SE) es tan amplia como que abarca a la comunidad científica, ya que si bien el trabajo se centra en el ámbito de una investigación concreta, no es menos cierto que en este caso el ámbito de aplicación es tan amplio que no cabe la realización de análisis alguno, razón por la cual en el marco del trabajo que se presenta, este modelo no puede ser analizado debido a la extensión y amplitud mencionadas.

Continuando con el contexto, nos encontramos con el modelo de tarea. Es aquí donde realmente cobra verdadera importancia la propuesta que formulamos en el presente trabajo. La tarea está perfectamente identificada: crear un SBC para anotación de secuencias fuera de su contexto genómico. Por tanto, la comunidad científica en este caso, que sustituye a una organización empresarial más concreta en casos específicos, se va a beneficiar mediante el desarrollo de esta tarea por las razones que motivan el trabajo, expuestas en los capítulos precedentes.

Desde el punto de vista del contexto, el tercer modelo que lo define es el de Agente. En este modelo se deben especificar todos los agentes involucrados desde el punto de vista de la rentabilidad y uso del sistema tanto desde el punto de vista físico como humano. En este sentido cabe especificar el uso de recursos remotos o locales con entidad propia, tal y como se estructuran desde el punto de vista de los sistemas multi-agente. Ciertamente, en el caso que nos ocupa el modelo de agentes carece de la entidad suficiente como para ser considerado, dado que no se producen interacciones entre diferentes entidades multi-agente, por lo que el modelo de comunicación entre supuestos agentes carece igualmente de sentido. Esto no es un motivo de invalidez del marco proporcionado por CommonKADS ya que es muy probable y así se espera, que en los desarrollos futuros estén involucrados diversos organismos o instituciones con intervención de diferentes usuarios en ubicaciones geográficamente dispersas o no que requerirán necesariamente la definición de modelos específicos de agentes, junto con las comunicaciones necesarias entre ellos.

Por tanto, a nivel conceptual y descartada la necesidad de descripción de un modelo de comunicación para el caso de nuestra tarea de anotación, el único modelo que queda es el de conocimiento, cobrando en nuestro caso una relevancia de envergadura.

Es en los modelos de tarea a nivel contextual y de conocimiento a nivel conceptual donde radica el énfasis del presente trabajo[39], describiéndose posteriormente con los detalles suficientes para la consecución del objetivo propuesto. Dentro del modelo de conocimiento, el énfasis se sitúa en la extracción del conocimiento y su estructuración para realizar los razonamientos pertinentes de cara a la consecución de la tarea propuesta de anotación de secuencias de ADN.

Finalmente, dentro del equipo se hace referencia al diseño necesario para materializar el SBC especificado, aquí se hace más bien referencia a temas muy próximos a la Ingeniería del Software donde se contemplan aspectos relacionados con el análisis, diseño e implementación.

Estos aspectos, aunque ciertamente importantes dentro del conjunto, son menos importantes que la tarea y el conocimiento que se consideran como relevantes desde la perspectiva del trabajo que se presenta y la necesidad identificada en la comunidad científica. No obstante, su inclusión se contempla desde el punto de vista de su necesidad de cara a la implementación de cualquier SBC.

4.2. Modelo de conocimiento

Uno de los retos de la Ingeniería del Conocimiento es encontrar estructuras para modelar el conocimiento de forma esquemática. CommonKADS ofrece un modelo de conocimiento compuesto por tres partes, denominadas **categorías de conocimiento**.

La primera categoría, el **conocimiento de dominio**, abarca el dominio específico del conocimiento y los tipos de información sobre los que se habla en la aplicación. El Capítulo 2 explica los conceptos básicos para la comprensión del dominio tratado en este trabajo, mientras que el Capítulo 3 describe y especifica el problema a ser tratado.

Este conocimiento está en el nivel más bajo del modelo de conocimiento y consta de dos elementos básicos: *esquema de dominio* y *Base de Conocimiento*.

El **esquema de dominio** es una descripción esquemática del conocimiento específico de dominio mediante la asociación de atributos y valores a determinadas instancias que ocurren en el dominio y que comparten características similares. Estas instancias se denominan **conceptos** y cada uno de sus **atributos** requiere un tipo para especificar los **valores** admitidos por este atributo. La Figura 4.2.a muestra el concepto *secuencia* y algunos de sus atributos y valores. Los conceptos pueden estar conectados a través de relaciones, como la descripta en la Figura 4.2.b, donde una secuencia codifica 0 o más *proteínas*. La dependencia entre conceptos es representada por relaciones lógicas, en el caso que nos ocupa, reglas. La Figura 4.2.c muestra un ejemplo de regla relativa al dominio analizado: *si una secuencia es*

un pseudogen entonces no puede codificar una proteína. El esquema de dominio puede poseer también subtipos (Figura 4.2.d) y puede ser descrito en diferentes niveles de abstracción, si bien abarcarlo en su plenitud desborda el ámbito de este trabajo. Nuestra intención consiste en destacar aquí esta circunstancia relativa a la posibilidad de poder utilizar subtipos.

La **Base de Conocimiento** (BC) contiene instancias de los tipos de conocimiento existentes en el esquema de dominio. Un ejemplo, es una BC formada por instancias del tipo reglas, citadas anteriormente.

Para comprender el dominio en el ámbito de nuestra aplicación es necesario conseguir esquematizarlo y crear una BC con su correspondiente información para posteriormente adquirir conocimiento sobre este dominio. Existen diferentes formas de obtener información respecto al dominio estudiado, como por ejemplo, a través de la literatura existente en relación a las anotaciones, experiencia laboral entre otras. En este trabajo el conocimiento que se tiene del dominio será complementado por el conocimiento de expertos en anotación de secuencias, empleando la entrevista como técnica clave de extracción del conocimiento (Sección 4.3).

El **conocimiento de inferencia** está en el segundo escalón de la pirámide y describe los pasos básicos necesarios para hacer uso del dominio de conocimiento. Este conocimiento está compuesto principalmente por las *inferencias*, los *roles del conocimiento* y las *funciones de transferencia*.

Las **inferencias** utilizan el conocimiento contenido en una BC para deducir nueva información a partir de una entrada. Pueden ser vistas como bloques para construir el razonamiento de la máquina y se describen a través de su entrada y salida, sin especificar el proceso interno utilizado para generar la salida. El Cuadro 4.1 presenta una lista de inferencias.

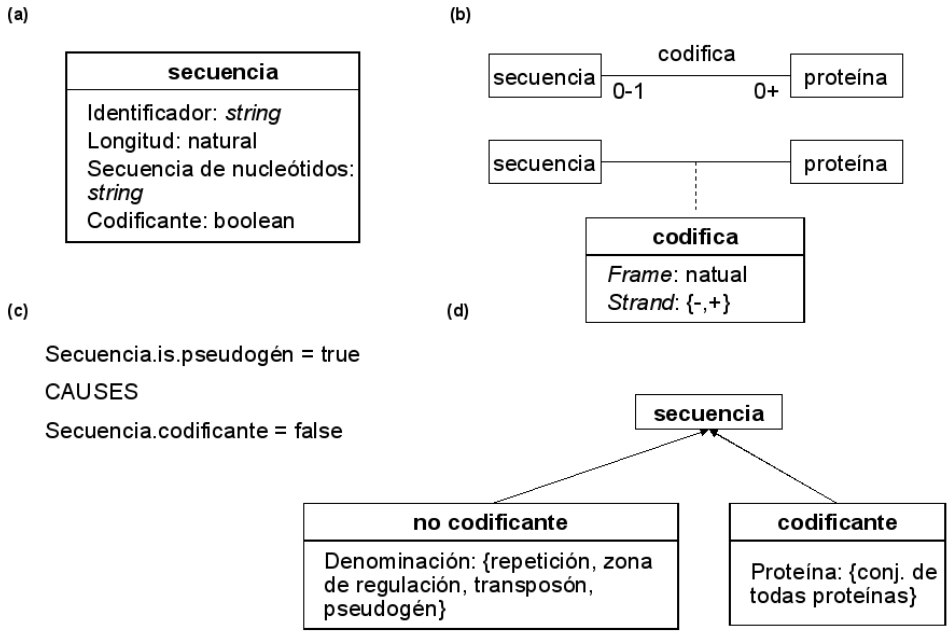

Figura 4.2: *Diferentes abstracciones del esquema de dominio. Ejemplo de la estructura de (a) concepto, (b) relación (c) regla y (d) subtipo.*

Catalogar	*Generar*	*Operacionalizar*
Criticar	*Proponer*	*Cubrir*
Modificar	*Verificar*	*Especificar*
Ordenar	*Comparar*	*Corresponder*
Asignar	*Agrupar*	*Clasificar*
Evaluar	*Predecir*	*Seleccionar*

Cuadro 4.1: *Lista de inferencias.*

La entrada/salida de la inferencia se estructura en función de los **roles de conocimiento**, es decir, objetos de datos que indican el razonamiento empleado en el proceso. El rol de conocimiento puede ser dinámico, cuando se trata de entradas o salidas de las inferencias en tiempo de ejecución, o estático, cuando es más o menos estable en el tiempo y especifica la colección del domino de conocimiento empleada para realizar la inferencia. En la Figura 4.3, *secuencia* y *candidatos* son roles dinámicos, mientras que *BD*, una BD perteneciente a la BC, es un rol estático.

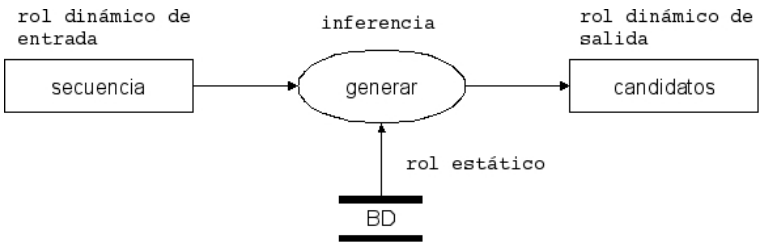

Figura 4.3: *Representación de los roles de conocimiento de la inferencia generar.*

		Iniciativa	
		Sistema	Externa
Información	Externa	*obtener*	*recibir*
	Interna	*presentar*	*proporcionar*

Cuadro 4.2: *Esquema de las funciones de transferencias con sus respectivas iniciativas y informaciones a transferir.*

La **función de transferencia** es una función que transfiere un ítem de información entre el proceso o agente encargado del razonamiento y el mundo exterior. Hay cuatro tipos de funciones de transferencia (Cuadro 4.2) donde cada uno se caracteriza por una iniciativa, que puede ser propia del sistema o externa, y por un elemento de información, interno o externo, a ser transferido. Las funciones de transferencia *obtener* y *recibir* son las más utilizadas en modelos de conocimiento[32].

La Figura 4.4 muestra una visión general del sistema propuesto en este trabajo, mostrando la relación entre las inferencias, los roles de inferencia y una función de transferencia que se ha considerado esencial en el esquema propuesto. La inferencia *generar* puede ser ejecutada por un número N de programas, lo que crea un diseño flexible que puede ser adaptado a otros *pipelines*. Cada uno de los N programas tiene como entrada el rol dinámico *secuencia* y consulta diferentes Bases de Datos (BD) (rol estático) existentes en la BC. La BC es compuesta por la unión de todas las BD y conjuntos de reglas existentes en el sistema. A continuación, la inferencia *selecionar* actúa sobre el rol dinámico *candidatos*, generado en el paso anterior, y elige los datos que estén conforme a las reglas de la BC (rol estático).

Estas reglas varían de acuerdo con el programa y el enfoque utilizado y se basan en los valores de algunos atributos de la salida de los programas, como, por ejemplo, el *e-value* o el *bitscore* de BLAST. La información obtenida después del filtrado por reglas, representada por *resultados*, es una lista de uno o más datos y sus respectivos atributos y valores. El conjunto de inferencias *generar* y *selecionar* se ejecuta para cada programa del proceso. Después de la ejecución de cada programa, se verifica (inferencia *verificar*) el contenido de los resultados y éstos son almacenados en la *BD de resultados*. En caso de que se cumplan determinadas reglas, como ocurre en *resultados* $N-1$ *verificado*, los resultados son utilizados para ejecutar otro programa con el objetivo de generar datos complementarios al análisis. Después de ejecutados todos los programas, la función de transferencia *obtener* recupera los resultados, en cuyo momento se realiza una comparación (inferencia *comparar*) de los resultados con base en reglas contenidas en la BC, generando como resultado final, el rol dinámico *proteína*.

La Figura 4.5 muestra la aplicación del esquema propuesto anteriormente para un sistema que contiene cuatro programas ($N = 4$): *Pfam scan*, BLASTx contra *Ensembl Protein* y *nr* y BLAST2GO. En esta figura está implícito que, en cada programa, las inferencias *generar* y *seleccionar* están asociadas respectivamente a una BD y a un conjunto de reglas determinado. BLASTx, por ejemplo, es ejecutado dos veces y, consecuentemente, para generar los datos se emplean dos BD de proteínas diferentes en contenido, pero con estructura similar, *Ensembl Protein* y *nr*, mientras que el mismo conjunto de reglas es usado para seleccionar los resultados obtenidos. BLAST2GO aparece en este esquema como un programa cuya ejecución depende del resultado de BLASTx contra *nr*. Es importante resaltar que los programas vinculados a los resultados de otros generan datos complementarios, o sea, datos que no son esenciales a la toma de decisiones. Estos datos además de aportar información complementaria, refuerzan la inferencia de la anotación, elevando su grado de fiabilidad.

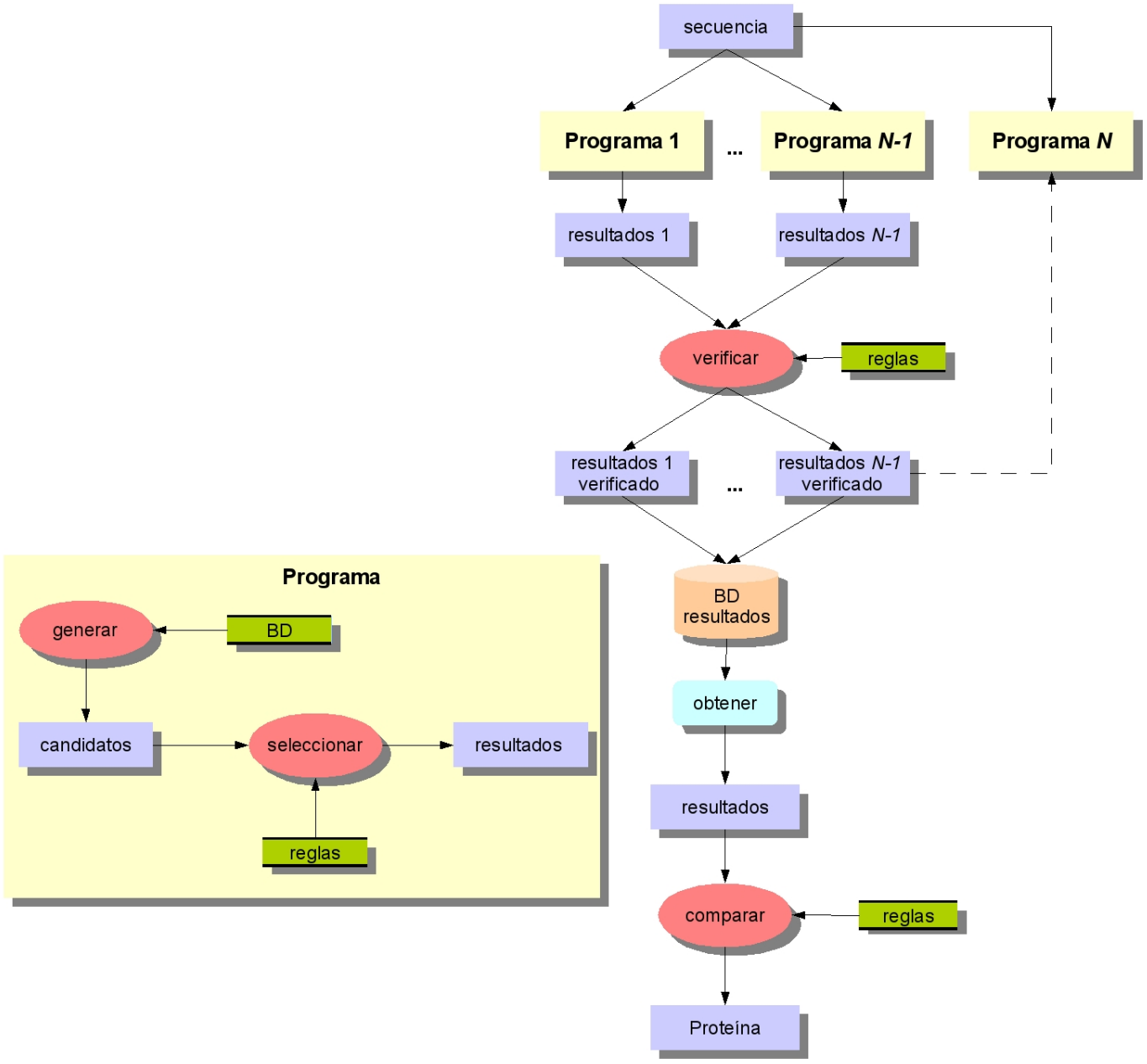

Figura 4.4: *Esquema general de inferencias, roles de inferencia y función de transferencia para aplicaciones de anotación de secuencia.*

El **conocimiento de tarea** especifica las metas de la aplicación y cómo estas metas pueden ser alcanzadas. Definir este conocimiento es muy importante para saber cómo diseñar

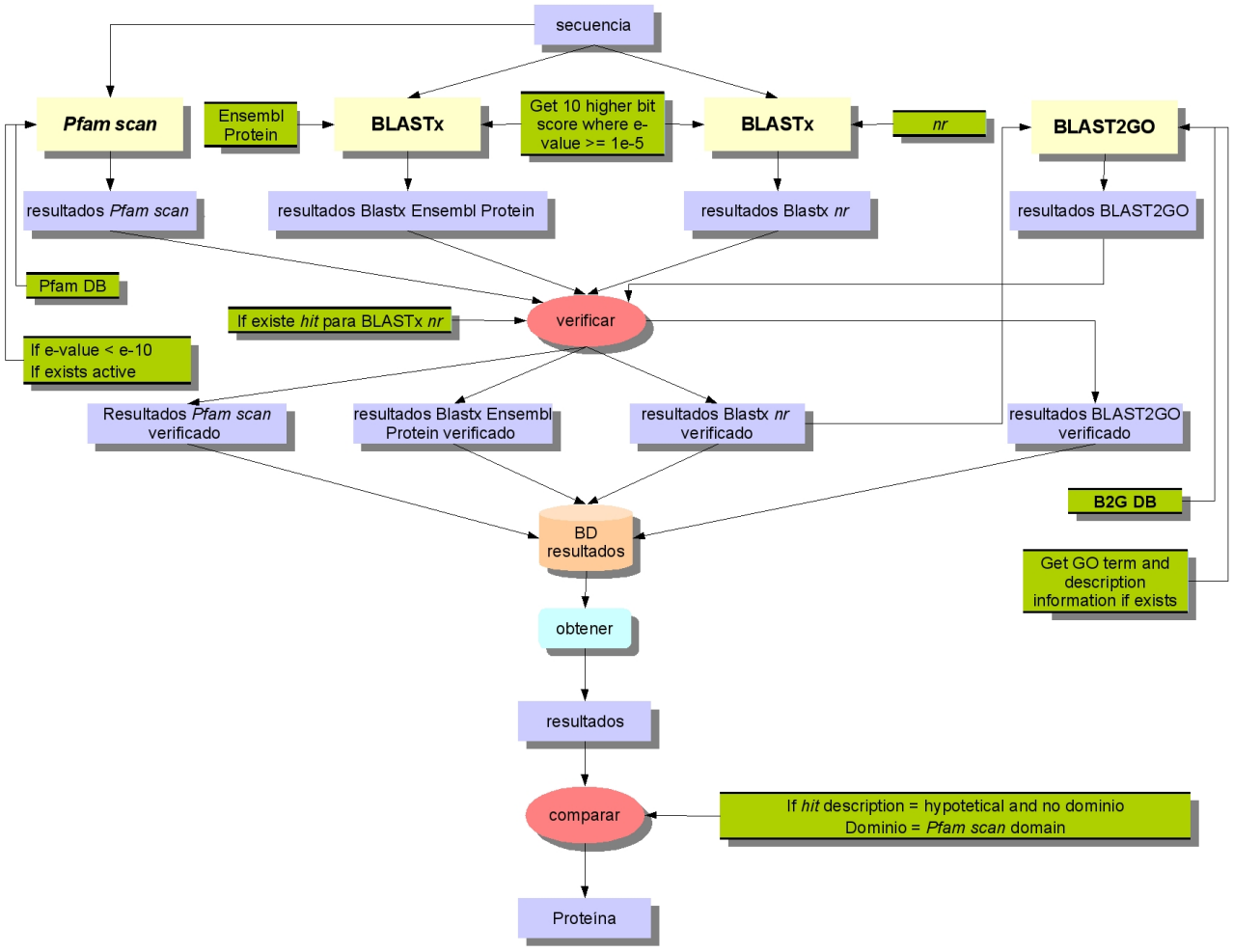

Figura 4.5: *Esquema de inferencias, roles de inferencia y función de transferencia para un pipeline de anotación compuesto por Pfam scan, BLASTx contra Ensembl Protein y contra nr y BLAST2GO.*

el sistema, por lo tanto, esta categoría es analizada con mayor profundidad a continuación, en la Sección 4.2.1.

El ejemplo de la Figura 4.6, esquematiza las tres categorías de conocimiento, vistas anteriormente, para una aplicación de anotación de secuencia. Como se puede observar, la tarea *clasificación* (explicada con más detalles en la Sección 4.2.1), localizada en el nivel superior, puede ser realizada a través de la invocación de las inferencias *generar* y *selecionar*.

La primera inferencia, va a generar las posibles proteínas candidatas encontradas en la BC, mientras la segunda seleccionará, con base en las reglas, los mejores resultados.

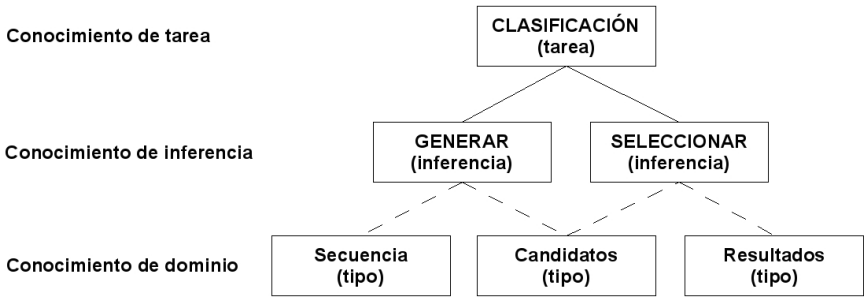

Figura 4.6: *Esquema de las categorías de conocimiento para una aplicación de anotación de secuencias.*

4.2.1. Conocimiento de tarea

El conocimiento de tarea es la categoría del conocimiento que describe los objetivos a ser alcanzados a través del conocimiento y las estrategias que serán empleadas para conseguirlos. Este conocimiento se describe de forma jerárquica *top-down* (Figura 4.7), donde las tareas del nivel superior se descomponen en tareas más pequeñas, que a su vez se vuelven a descomponer en tareas todavía menores. En el nivel más bajo de la descomposición las tareas se vinculan a inferencias y funciones de transferencia.

Este conocimiento tiene como elementos principales la *tarea* y el *método de tarea*. La **tarea** está relacionada con la pregunta "¿qué debe ser hecho?" y define una meta de razonamiento complejo especificando sus entradas y salidas en términos de roles funcionales. Cada tarea posee un **método de tarea** correspondiente que describirá cómo se realiza la tarea a través de una descomposición de subfunciones. El **método de tarea** contesta al interrogante "¿cómo se puede hacer?" y está compuesto por las subfunciones y por una estructura de control, que describe el orden en el que las funciones se llevan a cabo.

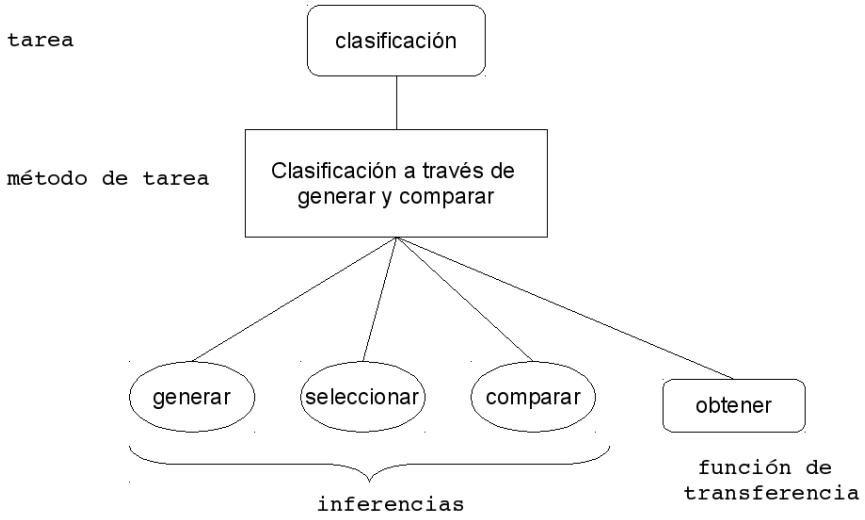

Figura 4.7: *Diagrama de la descomposición de la tarea clasificación.*

CommonKADS posee modelos de plantillas de tareas genéricas que pueden ser aplicados en diferentes dominios sin la necesidad de grandes adaptaciones y reutilizados independiente de la aplicación. Esto precisamente constituye una de las grandes ventajas de CommonKADS, que aprovechamos en este trabajo para diseñar el esquema de anotación de sequencias de ADN propuesto. Estas plantillas de tarea constituyen un modelo de conocimiento parcial en el cual se especifican las inferencias y el conocimiento de tarea y, según Pajares y Santos[32], han sido probadas en proyectos de ingeniería del conocimiento con el éxito suficiente para avalarlas.

Dentro de los modelos de tareas genéricas éstas se dividen en *analíticas* y *sintéticas* según cómo opera la tarea en el sistema, entendiendo por sistema como el objeto sobre el que actúa la tarea. En las **tareas analíticas** el sistema ya existe, pero no se le conoce completamente. Las tareas de análisis tienen como entrada algún dato procedente del sistema y producen como salida alguna caracterización del propio sistema. Por otro lado, la **tarea sintética** tiene como objetivo construir una descripción del sistema, puesto que éste todavía no existe. Las entradas de la tarea de síntesis suelen ser requerimientos que el sistema a construir debe

Tarea de análisis	
Tarea	**Descripción**
Clasificación	tarea que clasifica un objeto de acuerdo con la clase a la que éste pertenece. Esta tarea suele envolver objetos naturales, es decir, que no fueron hechos por el hombre, como por ejemplo, animales y plantas.
Diagnosis	tarea frecuentemente encontrada en el área de los sistemas técnicos, que tiene en cuenta el comportamiento del sistema. Difiere de la clasificación en que la salida deseada es un mal funcionamiento del sistema, que puede ser un fallo en un componente o en un estado del sistema entre otros. Un ejemplo es la avería de un coche o, en medicina, la detección de una enfermedad.
Valoración	tarea generalmente encontrada en el dominio financiero o de servicios. Su objetivo es caracterizar un caso en términos de decisión de clase (si/no/se necesita más información), por ejemplo, valorar si una persona está apta o no a recibir un préstamo.
Monitorización	tarea que analiza un sistema dinámico, en general a lo largo de su ejecución. Cada ciclo de monitorización puede ser visto como una pequeña valoración, con la diferencia que la salida de la monitorización no es una clase de decisión y sí indica si la conducta del sistema es normal o no.
Predicción	tarea que analiza el comportamiento de un sistema para construir una descripción del estado de este sistema en un momento futuro. La previsión del tiempo es un ejemplo de esta tarea.

Cuadro 4.3: *Descripción de las tareas de análisis.*

satisfacer.

Las tareas analíticas y de síntesis se subdividen en diferentes tipos de tareas de acuerdo con el tipo de problema a ser abordado. Esta clasificación aparece reflejada detalladamente en los Cuadros 4.3 y 4.4.

El catálogo de plantillas presentado en Schreiber y col.[39] contiene plantillas para todas las tareas citadas en este apartado a excepción de predicción y modelado. La descripción de las plantillas de tarea consta de cuatro elementos básicos. El primero es la **caracterización general** que describe las características típicas de la tarea, tales como el objetivo, la entrada y salida, la terminología empleada. El segundo, el **método por defecto**, se especifica en términos de roles, subfunciones y una descripción del control interno, en caso de que existan variaciones de este método, éstas son descritas en el apartado **variaciones del método**. Finalmente, en el **esquema típico de dominio** cada método asume la naturaleza del dominio de conocimiento subyacente.

Tarea de síntesis	
Tarea	**Descripción**
Diseño	tarea empleada cuando se desea construir un sistema físico, como por ejemplo, un coche, un ordenador, una placa eléctrica, etc.
Asignación	tarea donde se tiene dos conjuntos de objetos con los cuales se desea crear una asociación. Por ejemplo, asignar los aviones a las puertas de embarque. Esta tarea puede tener en cuenta restricciones (Aviones Iberia sólo pueden aterrizar en la Terminal 4) y preferencias (Boeing 747 deben colocarse en las puertas 35-40).
Planificación	tarea que comparte muchas características con el diseño, pero a diferencia del diseño, está relacionada con las actividades y sus dependencias de tiempo. Como por ejemplo, la planificación de viajes.
Scheduling	tarea que suele seguir a la planificación. Mientras la planificación entrega una secuencia de actividades, en el *scheduling* se debe asignar dicha secuencia de actividades a unos recursos durante cierto tiempo. La salida de esta tarea es una asignación entre actividades e intervalos de tiempo. Un ejemplo es *schedule* las clases de un departamento.
Modelado	tarea que consiste de construir una descripción abstracta de un sistema con el objetivo de explicar o predecir determinadas propiedades o fenómenos del mismo. Un propio modelado del conocimiento es un ejemplo de una tarea de modelado.

Cuadro 4.4: *Descripción de las tareas de síntesis.*

Como se ha mencionado previamente, la tarea a realizar en este trabajo es "anotar secuencias de ADN que se encuentran fuera de su contexto genómico". Esta tarea se encuadra perfectamente en la categoría identificada como tarea de clasificación en la metodología CommonKADS, puesto que existe la necesidad de clasificar el objeto secuencia como perteneciente a una determinada clase proteína. Por lo tanto, desde la perspectiva del trabajo que se plantea, se utilizará la plantilla de tarea de clasificación para modelar el conocimiento de tarea.

En la plantilla de tarea de clasificación la caracterización general consta de seis partes: el **objetivo**, donde se define la meta de la tarea; un **ejemplo típico**; la **terminología**, donde se especifican los términos que serán utilizados como el objeto que se desea clasificar, la **clase** a ser asignada a un grupo de objetos, los **atributos** que pueden ser observados o inferidos y la **característica** relacionada con cada atributo, es decir, un par atributo-valor para un objeto; el objeto de **entrada** y la clase de **salida**.

El **método por defecto** es la primera decisión a ser tomada. Éste puede ser **dirigido por los datos** o **por la solución**, siendo que el primero utiliza características iniciales del objeto para generar un conjunto de soluciones candidatas mientras el segundo empieza con el conjunto de todas las soluciones posibles y reduciendo este grupo con base a la información que se adquiere.

La **variación del método** enseña los diferentes caminos que pueden ser empleados para la solución del problema, cuando el método por defecto no puede ser utilizado. En este desarrollo, no existe ninguna variación del método, puesto que, todas las entradas pueden ser procesadas por el método por defecto.

A continuación, se presenta un esquema simplificado de la plantilla de clasificación de la tarea propuesta.

Plantilla de Clasificación para la tarea propuesta

Caracterización general

- **Objetivo:** establecer la proteína correspondiente a una secuencia dada. Esta clasificación está basada en las características del objeto secuencia obtenidas a través de alineamientos locales del objeto con secuencias existentes en diferentes bases de datos.
- **Ejemplo:** La secuencia de la Figura 4.8 es clasificada como Human proto-oncogene vav.
- **Terminología:**
  - **Objeto:** Objeto Secuencia.
  - **Clase:** Un elemento de la clase Proteína.
  - **Atributo:** *identificador*, *secuencia de nucléotidos*, *longitud de la secuencia*, *codificante*

- **Característica:**
  - el *identificador* siempre será un *string*, como por ejemplo, en el fichero FASTA de la Sección A.1, el identificador es: gi|24666374|ref|NM_079417.2|;
  - la *secuencia de nucleótidos* es un *string* de bases como se puede observar en la Figura 4.8;
  - la *longitud de la secuencia* es un número natural que muestra el número de pares de bases de la secuencia;
  - el *codificante* es un booleano, que presenta el valor verdadero en caso de que la secuencia codifique una proteína, es decir, cuando esta secuencia es un exón; y el valor falso cuando la secuencia no codifica ninguna proteína, pudiendo ser un pseudogén, un transposón, un zona de regulazión o una repetición.

- **Entrada:** Secuencia.
- **Salida:** Proteína.

```
GAATTCTGGGTATATTTCAGAGTGTCACTGCCGCCGTCTGCATATGGAGGAAGCTCACCC
ATCTCATAGTCTAGCTGGCCTGACTCCCCCAGCCCCCCAACTCCCCATGCCCAGGCCTGT
GTCGAGTGGGCGGAAGAAAGAGATGTAGATTCTGCATGGAAGGCGTGGGGTGGGGCTGGG
CTGCAGGTGCTCCCCCAGCTCCCCCCCGCCCCATGGCTCCTCCTCCTCCACCCCCTCTCA
GGGCGACAGTTACAGGAAAGAAGAGGAAGTGGTAGCACTAGCTGTCGCTCCACAGGCGAG
CAGGGCAGGCGTGCGGGCGGGTGGGTGGTGGAGGCTGCGAGGGTGCACGGCCGGCCCTGG
GCAGGGTAGCCATGGAGCTGTGGCGCCAATGCACCCACTGGCTCATCCAGTGCCGGGTGC
TGCCGCCCAGCCACCGCGTGACCTGGGATGGGGCTCAGGTGTGTGAACTGGCCCAGGCCC
TCCGGGATGGTGTCCTTCTGTGTCAGCTGCTTAACAACCTGCTACCCCATGCCATCAACC
TGCGTGAGGTCAAC
```

Figura 4.8: *Secuencia correspondiente a la proteína Proto-oncogene vav.*

Método por defecto

El método por defecto, como se ha mencionado anteriormente, será el único método empleado para solucionar el problema y, por lo tanto, no hay variaciones del mismo. Este método será dirigido a datos, pues a partir de una secuencia de nucleótidos se realizan

comparaciones a través de los programas con diferentes BD y se obtiene una lista de candidatos, que es filtrada tomando como base en determinadas reglas existentes en la BC, hasta encontrar una solución óptima.

El esquema de la tarea de clasificación de secuencias y su estructura de inferencias es similar al presentado en la Figura 4.4, si bien sin contener los roles estáticos.

4.3. Extracción del conocimiento

El modelado del conocimiento requiere que éste sea extraído de las fuentes que lo poseen. La **adquisición de conocimiento** es la tarea en que se obtiene el material necesario para construir una descripción más formal del problema. Esta tarea es el cuello de botella más importante a la hora de construir un SBC, puesto que esta actividad además de requerir una importante dedicación en tiempo, si el conocimiento no se extrae debidamente o contiene errores el sistema está abocado al fracaso.

La información a ser adquirida puede ser extraída de un experto en el tema o de otras fuentes como libros, manuales técnicos, etc. Los expertos son una importante fuente de conocimiento y poseen gran cantidad de información que, a veces, no ha sido catalogada previamente. No obstante, dicha información está estructurada de modo cognitivamente complejo, y parte de ella es tácita y pragmática[3], lo que dificulta el proceso de su extracción.

Para tener éxito en la adquisición del conocimiento, esta tarea debe ser centrada y estructura, pero a la vez debe estar abierta en lo posible. Además, se debe tener en cuenta el tipo de experto con quien se va a trabajar y buscar la técnica (o el conjunto de técnicas) de extracción del conocimiento más adecuada al dominio del problema.

Los resultados de esta tarea suelen ser algún tipo de dato mas o menos estructurado como diagramas, listas de términos, fórmulas, reglas informales, entre otros. En el trabajo

que proponemos, la extracción del conocimiento constituye una parte esencial del mismo debido a que en la comunidad científica existe un convencimiento generalizado sobre la falta de medios y técnicas para su extracción y posterior estructuración de un modo correcto.

4.3.1. Tipos de expertos

Conocer el **tipo de experto** que facilitará la información es esencial para extraer el conocimiento de forma eficaz. Aspectos relativos a su capacidad de comunicación, capacidad de verbalización y actitud con respecto al dominio van a influir en la manera con que el Ingeniero del Conocimiento debe proceder a la hora de extraer la información.

Existen tres tipos de expertos, a pesar, de que en la práctica éstos pueden poseer elementos de todos los tipos:

- Los **expertos más teóricos** están más acostumbrados a verbalizar su conocimiento y estructurarlo de forma lógica, pero no suelen aplicarlo en la práctica diaria.
- Los **expertos más prácticos** solucionan los problemas reales diariamente, pero no dominan tanto la teoría como los expertos más teóricos. Sin embargo, suelen emplear heurísticas en sus decisiones.
- Los **expertos solamente relacionados con la parte más práctica** realizan su labor de forma mecánica puesto que carecen de sustento teórico.

4.3.2. Técnicas de extracción del conocimiento

Las **técnicas de extracción del conocimiento** pueden ser divididas en **naturales** y **artificiales**, de acuerdo con la forma que el experto expresa su conocimiento. Cada técnica tiene sus ventajas e inconvenientes y debe ser empleada en diferentes etapas del modelado del conocimiento. Entre los métodos de adquisición del conocimiento más comunes se encuentran la entrevista, el protocolo, la observación directa, el escalonamiento, el agrupamiento de *clusters* y el emparrillado. La entrevista será descrita más adelante en esta sección mientras

que los demás métodos de adquisición del conocimiento se describen brevemente en el Cuadro 4.5.

Un método es descrito como natural si es adoptado por el experto cuando él, informalmente, expresa y demuestra su pericia. La **entrevista** y el análisis de protocolo son ejemplos de métodos naturales. Por otro lado, los métodos artificiales son aquellos empleados cuando el experto ejecuta una tarea artificial que proporciona conocimiento de determinadas maneras que no le suelen ser familiares. Técnicas como el escalonamiento, el agrupamiento de *clusters* y el emparrillado son ejemplos de métodos artificiales.

Los métodos de adquisición de conocimiento también pueden ser clasificados conforme a la manera como son realizados. Un método es denominado **manual** cuando el Ingeniero del Conocimiento interactúa diferentemente con el experto o con otras fuentes de información para extraer el conocimiento. Por otro lado, en un método **semiautomático** el Ingeniero del Conocimiento utiliza programas inteligentes o de aprendizaje **automático** para facilitar la extracción de reglas y heurísticas. La entrevista y el protocolo son métodos manuales, mientras que el escalonamiento, el agrupamiento de *clusters* y el emparrillado pueden ser implementados semiautomáticamente.

El método de extracción del conocimiento está directamente vinculado con el tipo de conocimiento que se desea adquirir, por lo tanto, para su eficiencia es esencial aplicar la técnica más adecuada al tipo de información del dominio. Los tipos de conocimiento pueden ser clasificados en:

- El **conocimiento procesal** está vinculado a cómo ejecutar una tarea dada. Este tipo de conocimiento está muy automatizado y, por lo tanto, el experto tiende a tener dificultad de verbalizarlo. La técnica más adecuada para extraerlo es la entrevista estructurada, donde se busca identificar rutinas y procedimientos empleados por el experto.

Nombre del método	**Descripción del método**	**Salida del método**	**Empleado para**	**¿Cuándo emplear?**
Análisis de protocolo	El experto verbaliza en voz alta su pensamiento mientras resuelve un problema	Protocolo	Generar especificación de tareas	En dominios en los que el conocimiento pueda expresarse de forma natural; en el análisis del razonamiento dinámico
Observación directa	Observar al experto mientras trabaja en sus tareas habituales en situaciones reales	Apuntes que serán discutidos a lo largo del proceso o posteriormente con el expertos	Conseguir una visión directa del trabajo del experto	En cualquier trabajo de adquisición del conocimiento que no sea totalmente de tipo conceptual
Escalonamiento	El experto y el ingeniero construyen una representación gráfica que relaciona dominio y elementos para la solución de problemas	Grafo bidimensional donde los nodos están conectados por aristas etiquetadas. Este grafo toma la forma de un árbol jerárquico	Construir jerarquías iniciales e informales	En fases iniciales de la exploración del dominio
Agrupamiento de *clusters*	Agrupar los elementos que están más próximos entre sí a través de una matriz de distancia	Árbol jerárquico, en donde aparecen los diferentes niveles de los agrupamientos y una escala que indica el valor de la distancia en que los elementos se agrupan	Adquirir conocimiento fuertemente jerarquizado de forma rápida	Cuando se trata de revelar la estructura de un dominio con lo cual no se está familiarizado
Emparrillado	Construcción de una matriz que relaciona ciertos elementos del dominio con el grado de ausencia o presencia de estos elementos de acuerdo a una escala de valores	Matriz	Crear esquema del dominio cuando no se está familiarizado con él	Cuando se trata de revelar la estructura de un dominio con lo cual no se está familiarizado

Cuadro 4.5: *Técnicas de extracción del conocimiento.*

- El **conocimiento declarativo** es aquel que puede ser verbalizado de forma sencilla. La entrevista no estructurada es el método ideal para buscar las heurísticas generales de este conocimiento.
- El **conocimiento semántico** está relacionado con el conocimiento de palabras y símbolos y de la organización cognitiva que tiene el experto. Incluye la memorización de vocabulario, conceptos hechos, definiciones y relaciones entre hechos. La observación directa es una de los métodos más adecuados para extraer este conocimiento.
- El **conocimiento episódico** se refiere al conocimiento de episodios o casos ocurridos anteriormente. Para conseguir extraer el conocimiento de situaciones análogas se debe aplicar métodos como el emparrillado o agrupamiento de *clusters*.

La entrevista

La **entrevista** es la técnica de extracción del conocimiento más usada, pues permite adquirir diferentes tipos de conocimiento y a distintos niveles del proceso de desarrollo del SBC, independiente del dominio de aplicación. Esta técnica no requiere un entrenamiento especial del Ingeniero del Conocimiento en cuanto al uso de la técnica en sí[3] y puede variar desde la entrevista uno-uno hasta formas más complejas con diversos expertos y/o ingenieros del conocimiento. La manera como el Ingeniero del Conocimiento entrevista al experto puede ser *estructurada* o *no estructurada*, dependiendo de como es conducida.

Entrevista no estructurada

La **entrevista no estructurada** no sigue ninguna planificación detallada y, por lo tanto, tiene pocas restricciones. A través de esta técnica el Ingeniero del Conocimiento puede conseguir una visión general sobre el tema y establecer una relación con el experto, lo que será favorable para futuras entrevistas. Además, este método permite al experto describir el

dominio a su manera, discutiendo tópicos que él considera importantes e ignorando aquellos que no le parecen tener tanto interés.

Sin embargo, la falta de estructura puede llevar a la ineficiencia del proceso, pues el experto puede sobrevalorar determinados tópicos en detrimento de otros, abarcando el dominio de forma irregular. Además, puede ser difícil de integrar los datos adquiridos, especialmente cuando provienen de diferentes expertos.

En la entrevista no estructurada se suelen formular preguntas abiertas, o sea, que necesitan de discusión y que no pueden ser contestadas con un valor numérico o un sí/no. Se aconseja que esta técnica sea empleada preferiblemente por la mañana, cuando el experto todavía no está cansado, y que tenga duración de una a tres horas como máximo. También es preferible realizar la entrevista en el ambiente de trabajo del experto, para facilitar que éste aporte casos, artículos, ejemplos, etc. Esta técnica suele ser aplicada en fases iniciales del modelado de conocimiento.

Entrevista estructurada

La **entrevista estructurada** sigue una estructura formal en la cual el Ingeniero del Conocimiento planifica y dirige la sesión con el objetivo de extraer el conocimiento específico utilizado por el experto para solucionar un determinado problema. Esta técnica prioriza la profundidad en lugar de la amplitud y se materializa en forma preguntas cerradas para alcanzar su meta.

Básicamente el método de la entrevista consiste en:

1. Solicitar que el experto describa brevemente la tarea a solucionar, incluyendo:
    - Un esbozo de la tarea y posibles soluciones para el problema;

Pregunta	Efecto
¿Por qué hace eso?	Convierte aserciones en reglas
¿Cómo lo hace?	Genera reglas de orden inferior
¿Cuándo lo hace? ¿Es cierta la regla siempre para este caso?	Revela la generalidad de la regla y puede generar otras reglas
¿Qué alternativas a la decisión/acción hay?	Genera más reglas
¿Qué pasa si no se da el caso de que?	Genera reglas para cuando ese supuesto no es aplicable
¿Puede decirme algo más acerca de?	Se usa para generar diálogos posteriores si el experto se queda en blanco

Cuadro 4.6: *Plantilla con preguntas para realizar una entrevista estructurada.*

- Una descripción de las variables que influencian en la elección de una posible solución;
- Una lista de las principales reglas que conectan las variables y las salidas.

2. Tomar cada una de las reglas descritas y preguntar cuándo es conveniente aplicarlas y cuándo no;

3. Repetir el paso 2 hasta que no se pueda obtener más información nueva.

Además de cumplir con los pasos del esquema anterior, el Ingeniero del Conocimiento puede seguir una plantilla con preguntas (Cuadro 4.6) para conseguir explorar al máximo el conocimiento del experto.

La entrevista estructurada genera como salida registros estructurados que son analizados con más facilidad que aquellos producidos por la entrevista no estructurada. Este método produce información relevante cuando se aplica en fases posteriores a la identificación del conocimiento y mientras la fase inicial de especificación del conocimiento; no obstante, es particularmente útil en la etapa de refinamiento del conocimiento, donde las BC deben ser rellenadas[39].

4.3.3. Directrices para la extracción del conocimiento en anotaciones de secuencias de ADN

Este trabajo tiene como objetivo comprender cómo el experto realiza el proceso de anotación y extraer las reglas y heurísticas aplicadas por él a lo largo de este proceso. El tipo de conocimiento a ser adquirido puede ser clasificado como procesal pues se refiere a cómo una tarea es ejecutada, a pesar de poseer algunas características declarativas, semánticas y episódicas.

La técnica empleada para extraer el conocimiento será exclusivamente la entrevista, más específicamente la entrevista estructurada, puesto que es el método que se adecua al tipo de conocimiento a ser extraído. Además es una técnica barata, que no requiere ningún entrenamiento especial por parte del entrevistador y que produce datos estructurados que pueden ser fácilmente convertidos en reglas y heurísticas. Otro punto importante es que la entrevista puede ser aplicada en diferentes etapas del modelado del conocimiento y dado que el Ingeniero del Conocimiento responsable por el desarrollo de la aplicación ya tiene conocimiento previo del dominio, se empleará este método con el objetivo de refinar el conocimiento, en lugar a comprender el dominio.

Los profesionales entrevistados poseen un perfil experto mixto con calidades tanto teóricas como prácticas y están bastante familiarizados con el dominio. La conclusión relativa a la extracción del conocimiento por medio de las entrevistas de las entrevistas se encuentran en el Apéndice B.

4.4. Conocimiento extraído a partir de las entrevistas

A partir de las entrevistas realizadas se puede tener una visión general con relación al proceso de anotación y también obtener las reglas específicas empleadas por cada especialista. Se observa que cada experto utiliza su propio *pipeline* de anotación, lo que hace necesario diseñar un modelo de anotación que sea flexible y que pueda ser modificado de acuerdo con

la necesidad y conocimiento del profesional. En la Figura 4.4 se esquematiza un modelo genérico para la anotación de secuencias de ADN capaz de adaptarse a diferentes *pipelines* de anotación. La Figura 4.5 representa la aplicación de esquema anterior a un *pipeline* real para la anotación de secuencias de ADN.

Básicamente, la anotación de secuencias de ADN puede ser vista como un problema de clasificación cuyos pasos (inferencias) principales son *generar* y *seleccionar*. La forma de abordar *generar/seleccionar* es utilizada para todos los programas y sus resultados pueden ser comparados entre si o utilizados como complemento de la información ya obtenida.

Comparando las entrevistas se puede percibir que:

- Diferentes caminos pueden llevar a las mismas conclusiones;
- BLAST está presente en todos los procesos, a pesar de que cada profesional utiliza diferentes *pipelines* para anotar sus secuencias;
- Los atributos de salida de los diferente programas son utilizados para tomar decisiones a lo largo del proceso de anotación. Las características de algunos de estos atributos son utilizadas en las reglas existentes en la BC a la hora de *seleccionar*. Por ejemplo, los expertos utilizan el atributo *e-value* para *seleccionar* resultados fiables entre los candidatos;
- Cada programa consultará diferentes BD. *Ensembl Protein* y *nr* son BD utilizadas por BLASTx, mientras Pfam DB es utilizada por *Pfam scan* y B2GO DB por BLAST2GO.

4.5. Análisis

En las secciones anteriores se ha adquirido conocimiento sobre el dominio de la aplicación a desarrollar y se ha modelado de forma general la tarea a ser ejecutada por el SBC. El SE aquí planteado y esquematizado se basa en reglas y éstas fueron adquiridas a través

de las entrevistas realizadas con los especialistas en el área de anotación de secuencias y, complementadas por el conocimiento propio previo en el área de la Bioinformática.

Para desarrollar el sistema ilustrado en la Figura 4.5 es necesario definir los componentes básicos de este SBC. Como ilustra la Figura 4.9, un SE basado en reglas típico se compone de:

- **Base de conocimiento:** contiene el conocimiento de dominio necesario para resolver los problemas codificados en forma de reglas.
- **Interfaz de usuario**: mecanismo que permite la comunicación entre el usuario y el sistema.
- **Medio de explicación**: describe al usuario la estructura de razonamiento utilizada por el sistema al llevar a cabo una inferencia.
- **Memoria activa**: colección de informacion utilizada por el sistema para decidir qué reglas deben ser aplicadas. Cuando el sistema es iniciado, la memoria activa normalmente incluye el dato de entrada. A lo largo de la ejecución del sistema esta memoria puede ser empleada para almecenar conclusiones intermedias y cualquier otra información necesaria.
- **Mecanismo de inferencia:** hace inferencias aplicando las reglas y sus prioridades.
- **Agenda:** listado de prioridades asignadas a la reglas, creado por el mecanismo de inferencia, cuyos patrones satisfacen a los hechos u objetos de la memoria activa.
- **Medio para la adquisición de conocimiento:** vía automática que posibilita al usuario introducir nuevo conocimiento al sistema.

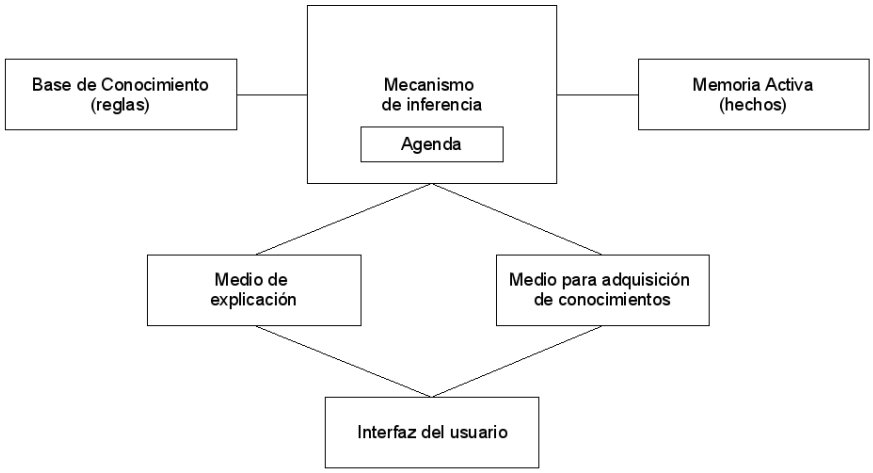

Figura 4.9: *Estructura de un Sistema Experto basado en reglas.*

Componente	**Descripción**	**Formato**
Pfam DB	Base de Datos de modelos de Markov de Pfam	MySQL
nr	Base de Datos no redundante de proteínas	Sigue el formato de las Bases de Datos de BLAST, generado por el programa denominado *formatd*
Ensembl Protein	Base de Datos de proteínas de Ensembl	Sigue el formato de las Bases de Datos de BLAST, generado por el programa denominado *formatd*
B2GO DB	Base de Datos de BLAST2GO	MySQL
Reglas	Colección de reglas	Sentencias de control condicional (*if-then-else*)

Cuadro 4.7: *Componentes de la Base de Conocimiento para el Sistema Experto descrito en la Figura 4.5.*

El SBC planteado en este trabajo tiene como **interfaz de usuario** la terminal del ordenador, donde el usuario puede definir los parámetros y ficheros de entrada a ser utilizados. En el futuro, se pretende desarrollar una intefaz web, pero este plan está condicionado por la disponibilidad de los recursos.

Este sistema posee una BC formada por BD externas, originarias de BD públicas, y una colección de reglas formadas con base en el conocimiento de los especialistas en anotación. El Cuadro 4.7 describe cada componente de la BC y su respectivo formato.

La **memoria activa** no es almacenada en una BD física, se utiliza una estructura de datos compuesta por *hashes* y *arrays* para simular virtualmente la BD y guardar tanto los datos de entrada como los intermedios. Esta memoria se representa por *BD resultados*, Figura 4.5.

El principal **mecanismo de inferencia** del SBC, representado por la inferencia *comparar* en la Figura 4.5, se alimenta de reglas de la BC y sus prioridades (**agenda**) para inferir la anotación final, *Proteína*. Sin embargo, cada programa utilizado posee un mecanismo de inferencia asociado que será empleado para inferir los mejores resultados.

El **medio de explicación** de los pasos de inferencia se genera en el *log* del sistema, pero no se enseña directamente al usuario. Por otro lado, el **medio para la adquisición de conocimiento**, por ser una función opcional, no está presente en este sistema, pero puede ser añadido posteriormente, dada la modularidad prevista en el mismo.

La arquitectura del sistema fue deseñada como un conjunto de *pipelines* que pueden ser ejecutados en paralelo cuyos resultados son analizados después de la finalización de la ejecución de todos los programas. Esta arquitectura puede ser vista como un río que posee varios afluentes que desembocan en el mismo punto. Es importante tener en cuenta que el *pipeline BLAST2GO* depende de la finalización del *pipeline BLASTx nr*. Esta relación por lo tanto puede ser vista como un *pipeline*. La Figura 4.10 ilustra de forma resumida la arquitectura propuesta para el sistema de la Figura 4.5.

4.6. Diseño

La Figura 4.11 muestra el diseño detallado del sistema para una entrada en formato FASTA. Como se puede observar en esta figura, la entrada es utilizada por cuatro *pipelines* diferentes: *BLASTx Ensembl Protein*, *BLASTx nr*, *BLAST2GO* y *Pfam scan*.

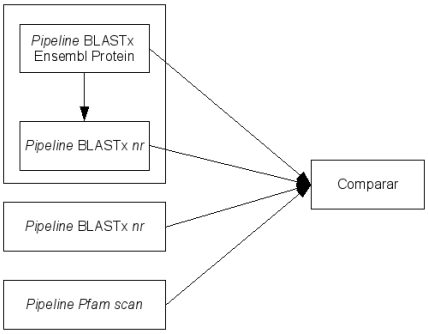

Figura 4.10: *Esquema de la arquitectura del sistema para el ejemplo de la Figura 4.5.*

Los *pipelines* de *BLASTx Ensembl Protein* y *BLASTx nr* son similares y utilizan los mismos parámetros (definidos en la BC). La única diferencia entre ellos es que el primero hace un BLAST contra *Ensembl Protein*, mientras el segundo contra *nr*. A continuación, el resultado obtenido por cada BLAST es *parseado* a través del script *best_hit_info.pl*, que busca los mejores *hits* obtenidos con base da reglas de la BC. Los resultados de *BLASTx nr* son enviados a *pipeline BLAST2GO*, en caso de que éstos posean algún *hit*. El BLAST2GO obtiene los términos GO y sus descricpiones relacionados a los *hits* obtenidos anteriormente. Esta información será utilizada para reforzar la asignación de las características de la secuencia de ADN, además de complementar su descripción.

El *pipeline de Pfam scan* empieza con la traducción de la secuencia de nucleótidos del FASTA a proteína. Como se ha explicado anteriormente en la Sección 2.5, esta traducción se hace para la secuencia original (*strand* positiva) y su complemento (*strand* negativa) teniendo en cuenta los tres marcos de lectura posibles (1, 2 y 3), generando consecuentemente seis ficheros en formato FASTA. Estos ficheros son la entrada del programa *Pfam scan* que los compara con los modelos existentes en la BD de modelos ocultados de Markov (Pfam DB). A continuación, los resultados de *Pfam scan* son evaluados por el *script get_pfam_info.pl* que, con base en reglas, busca los mejores resultados y la información asociada a ellos.

Librería	Descripción
Getopt::Long	Procesador extendido de opciones para línea de comando
Config::IniFiles	Módulo para la lectura de ficheros de configuración .ini
FindBin	Localiza la ruta completa del directorio donde el *script* está siendo ejecutado, permitiendo el uso de rutas relativas.
File::Basename	*Parsea* la ruta de ficheros, directorios y obtiene sufijos.
LWP::Simple	Interfaz simplicada para LWP que permite recuperar páginas *web*.
MySQL	Interfaz para Bases de Datos mSQL y MySQL
DBI	Interface de Base de Datos consistente e independiente de la Base de Datos que se esté usando en ese momento
Bio::SeqIO	Módulo de BioPerl capaz de manejar ficheros en formato FASTA, EMBL, gb, entre otros
Bio::SearchIO	Módulo de BioPerl para *parsear* ficheros generados por BLAST, FASTA, entre otros programas

Cuadro 4.8: *Componentes de la Base de Conocimiento para el Sistema Experto descrito en la Figura 4.5.*

Cuando todos los *pipelines* han finalizado, los resultados (*Best hit Blastx Ensembl Protein info*, *Best hit Blastx nr info*, *B2GO info* y *Pfam scan info*) son comparados a través de *comparar_resultados.pl*. Este *script parsea* los resultados y busca, con base en reglas, palabras claves en cada uno de ellos. Estas palabras son comparadas entre sí y posteriormente utilizadas para inferir la anotación más adecuada (fichero *Proteína*). Además de un fichero ASCII conteniendo la anotación asignada por el sistema, se crea un fichero de *log* que describe los pasos realizados hasta llegar a la decisión tomada.

4.7. Implementación

El lenguaje de implementación del SBC es PERL, lenguaje de programación ampliamente utilizado por los Bioinformáticos debido a las facilidades ofrecidas para la manipulación de datos biológicos. La herramienta BioPerl es una de estas facilidades, pues posee un conjunto de librerías desarrolladas específicamente para *parsear* y analizar ficheros con formatos bioinformáticos. El Cuadro 4.8 lista las librerías utilizadas para la implementación del SBC y sus funcionalidades.

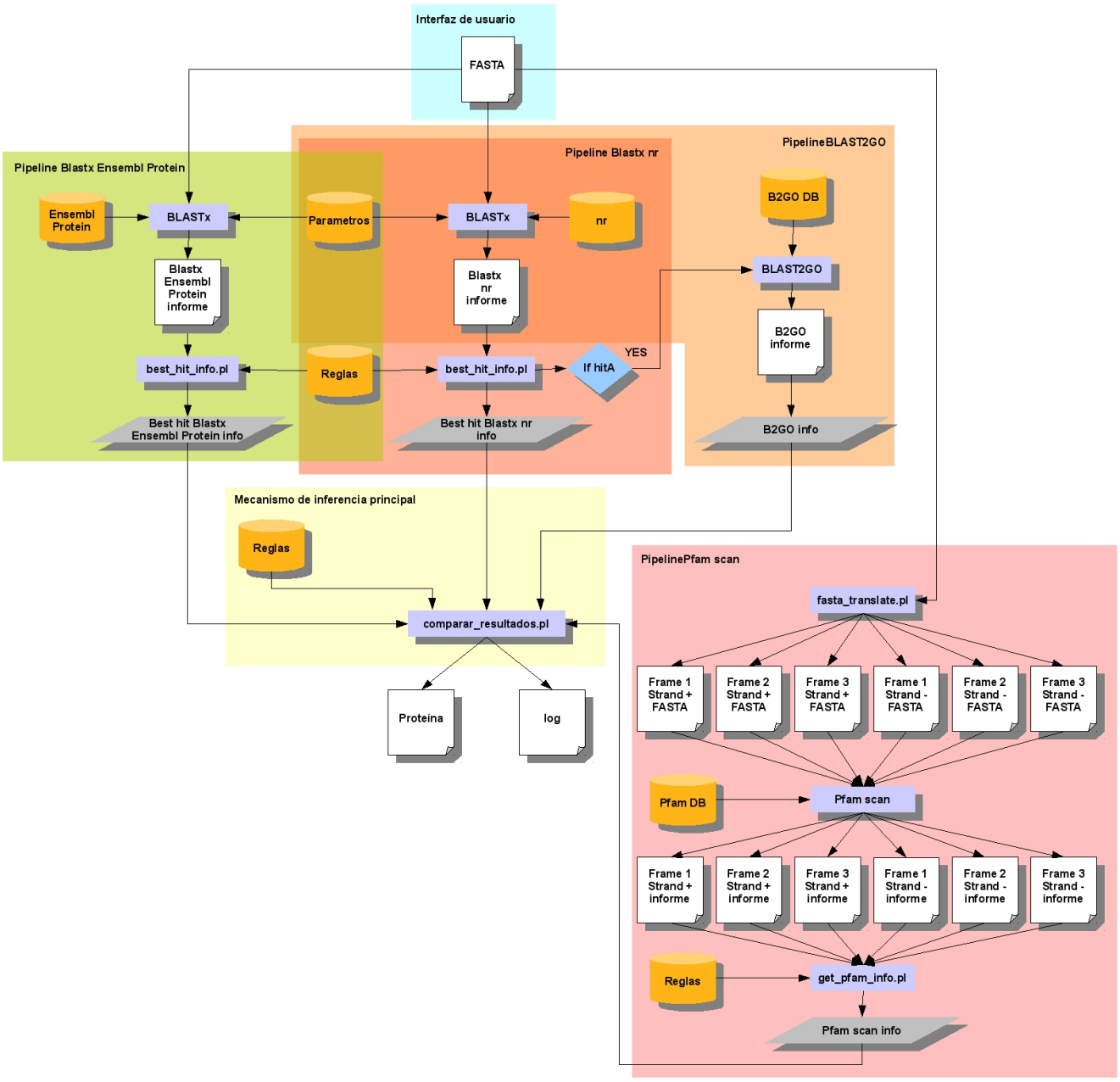

Figura 4.11: *Diseño del SBC propuesto en la Figura 4.5.*

Además de las librerías de PERL y de los *scripts* desarrollados, se han utilizado programas bioinformáticos (Sección 2.5) para procesar los datos: BLAST, Pfam scan y BLAST2GO. Los resultados obtenidos con esta implementación se describen en el Apéndice C.

Capítulo 5

Conclusión y trabajo futuro

Los avances tecnológicos de los últimos años han posibilitado el aumento de la producción de datos biológicos. Procesar, analizar, almacenar y utilizar estos datos eficaz y eficientemente es una tarea laboriosa. Este hecho puede constatarse en la anotación de genes, una de las tareas más desafiantes de la genómica, que consiste en identificar y caracterizar los genes de una porción de ADN.

La anotación de los genes idealmente debe ser llevada a cabo manualmente por un especialista que tiene conocimientos biológicos y bioquímicos para estudiar cada caso por separado e inferir las características adecuadas a cada gen encontrado. No obstante, la gran cantidad de datos genómicos a ser procesada imposibilita que esta tarea dependa de la intervención humana, lo que genera la necesidad de procesos automáticos de anotación.

La anotación automática evita el cuello de botella causado por la dependencia del especialista, pero no es tan fiable y exacta como la anotación manual. Actualmente, existen diferentes *pipelines* de anotación automática que se sirven de varios programas Bioinformáticos para alcanzar su objetivo. Estos programas muchas veces predicen erróneamente algunos genes o les asignan descripciones equivocadas. Como es una práctica habitual volcar los resultados de las anotaciones en Bases de Datos públicas para que puedan ser utilizadas en anotaciones futuras, al anotar una secuencia de forma errónea, este error será propagado a

las futuras anotaciones.

Este problema puede ser minimizado a través de la curación manual de los resultados obtenidos automáticamente, sin embargo, este proceso no elimina el cuello de botella que hace inviable la anotación manual. Una posible solución para este problema consiste en crear Sistemas Expertos capaces de emular el razonamiento del anotador en pasos clave de la anotación.

Sistemas Basados en Conocimiento para anotación, como el *pipeline* de anotación de Ensembl, han sido desarrolados direccionados a la anotación de genomas de euraciotas. No obstante, estos sistemas realizan la anotación de secuencias dentro de un contexto genómico, no siendo capaces de anotar secuencias aisladas de este contexto. Por tanto, crear un Sistema Experto para la anotación de secuencias independiente del contexto genómico puede elevar la calidad de los datos a ser insertados en las Bases de Datos públicas y, consecuentemente, contribuir para la mejora de las anotaciones futuras.

El desarrollo de Sistemas Expertos es una labor compleja que exige el conocimiento del dominio de aplicación del sistema y del proceso de razonamiento del especialista para la solución del problema. Metodologías para el modelado de Sistemas Expertos, como CommonKADS, auxilian el desarrollo de tales sistemas y pueden ser aplicadas en diferentes ámbitos. La aplicación de CommonKADS en el campo de la Bioinformática no había sido descrita anteriormente, hasta donde hemos podido constatar, lo que hace de este trabajo el primero en proponer el empleo de dicha metodología para solucionar un problema en este ámbito. Gracias al esfuerzo realizado en el diseño del Sistema Experto propuesto en este trabajo, se ha podido constatar que CommonKADS puede extenederse a diferentes problemas bioinformáticos, como la clasificación de secuencias provenientes de metagenomas o bien la predicción de la estructura tridimensional de proteínas. Constituyen de este modo el punto de partida para futuros trabajos de investigación en Bioinformática.

Este trabajo propone el diseño de un sistema de anotación basado en reglas capaz de anotar secuencias de ADN independiente de su contexto genómico utilizando el modelado CommonKADS y la entrevista como técnia de extracción del conocimiento. A través de esta metodología se ha creado un marco general para este problema, utilizando la tarea de clasificación como el elemento fundamental. Además, se ha logrado un alto nivel de generalización en el modelo diseñado, lo que permite que éste sea empleado con diferentes combinaciones de programas Bioinformáticos y para cualquier tarea de anotación de secuencias.

Con este trabajo se pretende resaltar la importancia de la aplicación de técnicas metodológicas para el desarrollo de sistemas en el área que nos compete. Estas técnicas proporcionan las herramientas necesarias para organizar el conocimiento de manera que éste pueda ser entendido por todo el personal involucrado en el desarrollo del sistema. Por otro lado, las técnicas de extración de conocimiento constituyen un poderoso medio de obtención de información esencial para la comprensión del dominio de la aplicación.

El conocimiento obtenido y estructurado a través de las técnicas descritas ha posibilitado la implementación del sistema ilustrado en la Figura 4.5. Este sistema ha sido utilizado para una aplicación real, y habiéndose obtenido resultados satisfactorios (Apéndice C). Sin embargo, se cree que se puede mejorar el mecanismo de inferencia añadiéndole más reglas, y por tanto, se pretende realizar más entrevista. Se pretende también optimizar el sistema y, en el futuro, desarrollar una intefaz web, pero este plan está condicionado por la disponibilidad de los recursos. Además, se ha modelado el sistema para la anotación funcional, enfocándose en inferir las proteínas que codifican cada secuencia, pero este modelado puede ser extendido a la anotación estructural. En este caso, este trabajo puede aportar conceptos y directrices que facilitarán el modelado de este nuevo enfoque.

Bibliografía

[1] 454 Sequencing. Products & solutions - system benefits: 454 life sciences, a roche company. http://454.com/products-solutions/system-benefits.asp. Acceso: 17 de marzo de 2011.

[2] 454/Roche. Newbler, 2009. http://www.454.com/.

[3] Alonso, A., and Guijarro, B. *Ingeniería del Conocimiento. Aspectos Metodológicos.*, 1 ed. Pearson Prentice Hall, 2004.

[4] Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. Gapped blast and psiblast: a new generation of protein database search programs. *NUCLEIC ACIDS RESEARCH 25*, 17 (September 1997), 3389–3402.

[5] Bairoch, A., and Apweiler, R. The swiss-prot protein sequence data bank and its supplement trembl in 2000. *Nucleic Acids Research 28* (2000), 45–48.

[6] Bakke, P., Carney, N., DeLoache, W., Gearing, M., Ingvorsen, K., Lotz, M., McNair, J., Penumetcha, P., Simpson, S., Voss, L., Win, M., Heyer, L. J., and Campbell, A. M. Evaluation of three automated genome annotations for *Halorhabdus utahensis*. *PLoS ONE 4*, 7 (07 2009), e6291.

[7] Barrett, S. J. Intelligent bioinformatics: The application of artificial intelligence techniques to bioinformatics problems: John wiley sons ltd., chichester, UK, keedwell, edward and narayanan, ajit, 2005, 280 p., hardcover, ISBN 0-470-02175-6. *Genetic Programming and Evolvable Machines 7*, 3 (Oct. 2006), 283–284.

[8] Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., and Wheeler, D. L. Genbank. 34–38. http://www.ncbi.nlm.nih.gov/genbank.

[9] Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.-C., Estreicher, A., Gasteiger, E., Martin, M., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S., and Schneider, M. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic Acids Research 31* (2003), 365–370.

[10] Bryson, K., Loux, V., Bossy, R., Nicolas, P., Chaillou, S., van de Guchte, M., Penaud, S., Maguin, E., Hoebeke, M., Bessières, P., and Gibrat, J.-F. F. AGMIAL: implementing an annotation strategy for prokaryote genomes as a distributed system. *Nucleic acids research 34*, 12 (July 2006), 3533–3545.

[11] Center for Information Biology and DNA Data Bank of Japan. Ddbj. http://www.ddbj.nig.ac.jp.

[12] Chevreux, B., Wetter, T., and Suhai, S. Genome sequence assembly using trace signals and additional sequence information. In *Computer Science and Biology: Proceedings of the German Conference on Bioinformatics (GCB) 99* (1999), pp. 45–46. http://chevreux.org/projects_mira.html.

[13] Claverie, J.-M., and Notredame, C. *Bioinformatics for Dummies*. Wiley Publishing Inc, 2007.

[14] Conesa, A., Götz, S., García-Gómez, J., Terol, J., Talón, M., and Robles, M. Blast2go: A universal tool for annotation, visualization and analysis in functional genomics research. *Bioinformatics 21* (2005), 3674–3676. http://www.blast2go.org.

[15] Curwen, V., Eyras, E., Andrews, T. D., Clarke, L., Mongin, E., Searle, S. M., and Clamp, M. The ensembl automatic gene annotation system. *Genome Res 14*, 5 (May 2004), 942–950.

[16] Edwards, D., Stajich, J. E., and Hansen, D. *Bioinformatics tools and applications*. Springer, 2009.

[17] EMBL-EBI. The embl nucleotide sequence database statistics. http://www.ebi.ac.uk/embl/Services/DBStats/. Acceso: 17 de marzo de 2011.

[18] Ensembl. Ensembl pipeline configuration. http://cvs.sanger.ac.uk/cgi-bin/viewvc.cgi/ensembl-pipeline/?root=ensembl.

[19] Finn, R., Mistry, J., Tate, J., Coggill, P., Heger, A., Pollington, J., Gavin, O., Gunesekaran, P., Ceric, G., Forslund, K., Holm, L., Sonnhammer, E., Eddy, S., and Bateman, A. The Pfam protein families database. *Nucleic Acids Res 38*, Database issue (2010), D211–222. http://pfam.sanger.ac.uk.

[20] Giarratano, J., and Riley, G. *Sistemas expertos. Principios y programación*. International Thomson, 1998.

[21] Gish, W. Wu-blast, 1996-2004. http://blast.wustl.edu.

[22] Gouret, P., Vitiello, V., Balandraud, N., Gilles, A., Pontarotti, P., and Danchin, E. Figenix: Intelligent automation of genomic annotation: expertise integration in a new software platform. *BMC Bioinformatics 6* (2005), 198. http://www.up.univ-mrs.fr/evol/figenix.

[23] Green, P. Phrap, 1999. http://phrap.org.

[24] Griffiths-Jones, S. Pfam scan. ftp://ftp.sanger.ac.uk/pub/databases/Pfam/Tools/OldPfamScan/pfam_scan.pl.

[25] Holland, R. C. G., Down, T., Pocock, M., Prlic, A., Huen, D., James, K., Foisy, S., Drager, A., Yates, A., Heuer, M., and Schreiber, M. J. BioJava: an Open-Source framework for bioinformatics. *Bioinformatics 24*, 18 (Aug. 2008), btn397–2097.

[26] Huang, X., and Madan, A. Cap3: A dna sequence assembly program. *Genome Research 9* (1999), 868–877. http://seq.cs.iastate.edu/cap3.html.

[27] Hubbard, T. J., Aken, B. L., Beal, K., Ballester, B., Caccamo, M., Chen, Y., Clarke, L., Coates, G., Cunningham, F., Cutts, T., Down, T., Dyer, S. C., Fitzgerald, S., Fernandez-Banet, J., Graf, S., Haider, S., Hammond, M., Herrero, J., Holland, R., Howe, K., Howe, K., Johnson, N., Kahari, A., Keefe, D., Kokocinski, F., Kulesha, E., Lawson, D., Longden, I., Melsopp, C., Megy, K., Meidl, P., Ouverdin, B., Parker, A., Prlic, A., Rice, S., Rios, D., Schuster, M., Sealy, I., Severin, J., Slater, G., Smedley, D., Spudich, G., Trevanion, S., Vilella, A., Vogel, J., White, S., Wood, M., Cox, T., Curwen, V., Durbin, R., Fernandez-Suarez, X. M., Flicek, P., Kasprzyk, A., Proctor, G., Searle, S., Smith, J., Ureta-Vidal, A., and Birney, E. Ensembl 2007. *Nucleic Acids Research 35*, Database issue (Jan. 2007). http://www.ensembl.org.

[28] Kanz, C., Aldebert, P., Althorpe, N., Baker, W., Baldwin, A., Bates, K., Browne, P., van den Broek, A., Castro, M., Cochrane, G., Duggan, K., Eberhardt, R., Faruque, N., Gamble, J., Diez, F., Harte, N., Kulikova, T., Lin, Q., Lombard, V., Lopez, R., Mancuso, R., McHale, M., Nardone, F., Silventoinen, V., Sobhany, S., Stoehr, P., Tuli, M., Tzouvara, K., Vaughan, R., Wu, D., Zhu, W., and Apweiler, R. The embl nucleotide sequence database. *Nucleic Acids Research 33* (January 2005), 29–33. http://www.embl.de.

[29] Lehninger, A., Nelson, D. L., and Cox, M. M. *Lehninger Principles of Biochemistry*, fifth edition ed. W. H. Freeman, June 2008.

[30] NCBI. Just the Facts: A Basic Introduction to the Science Underlying NCBI Resources. National Center for Biotechnology Information, Marzo 2004. http://www.ncbi.nlm.nih.gov/About/primer/bioinformatics.html.

[31] NCBI-GenBank. Genetic sequence data bank. ftp://ftp.ncbi.nih.gov/genbank/gbrel.txt. Acceso: 17 de marzo de 2011.

[32] Pajares, G., and Santos, M. *Inteligencia Artificial e Ingeniería del Conocimiento.* Ra-Ma, Librería y Editorial Microinformática, 2005.

[33] Pearson, W. R., and Lipman, D. J. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America 85*, 8 (Apr. 1988), 2444–2448. http://fasta.bioch.virginia.edu/fasta_www2/fasta_list2.shtml.

[34] Pevsner, J. *Bioinformatics and functional genomics.* Wiley-Blackwell, 2009.

[35] Potter, S. C., Clarke, L., Curwen, V., Keenan, S., Mongin, E., Searle, S. M., Stabenau, A., Storey, R., and Clamp, M. The ensembl analysis pipeline. *Genome Research 14*, 5 (May 2004), 934–941.

[36] Robinson, T. R. *Genetics for Dummies.* Wiley Publishing Inc, 2005.

[37] Romem, Y., Pulka, A., Dunstan, N., Petrică, V., Fan, X., Chen, X., Wang, Y., Narayanan, R. G., Klinger, A., Kronberger, T., Schaler, M., Schürz, B., Stohl, K., Sayed, H. E., Gabbar, H. A., Miyazaki, S., Zarandi, M. H. F., Avazbeigi, M., Anssari, M. H., Ganji, B., Aniba, M. R., Thompson, J. D., Rafea, A., Hu, Y., Wang, J., Li, P., Riahi-Madvar, H., and Ayyoubzadeh, S. A. *Expert Systems.* InTech, January 2010.

[38] Rust, A. G., Mongin, E., and Birney, E. Genome annotation techniques: new approaches and challenges. *Drug Discov Today 7*, 11 Suppl (2002), S70–6.

[39] Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., and Wielinga, B. *Knowledge Engineering and Management: The CommonKADS Methodology.* MIT Press, Cambridge, MA, 2000.

[40] Siegfried, D. R. *Biology for Dummies*. Wiley Publishing Inc, 2001.

[41] Slama, P., Filippis, I., and Lappe, M. Detection of protein catalytic residues at high precision using local network properties. *Bioinformatics 9* (December 2008).

[42] Smit, A. F. A., Hubley, R., and Green, P. RepeatMasker. http://www.repeatmasker.org.

[43] Sonnhammer, E., Eddy, S. R., Birney, E., Bateman, A., and Durbin, R. Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic Acids Res 26*, 1 (1998), 320–322.

[44] Stajich, J. E., Block, D., Boulez, K., Brenner, S. E., Chervitz, S. A., Dagdigian, C., Fuellen, G., Gilbert, J. G., Korf, I., Lapp, H., Lehväslaiho, H., Matsalla, C., Mungall, C. J., Osborne, B. I., Pocock, M. R., Schattner, P., Senger, M., Stein, L. D., Stupka, E., Wilkinson, M. D., and Birney, E. The bioperl toolkit: Perl modules for the life sciences. *Genome Research 12*, 10 (Octuber 2002), 1611–8.

[45] Stein, L. Genome annotation: from sequence to biology. 493–503.

[46] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics 25*, 1 (May 2000), 25–9. http://www.geneontology.org.

[47] The UniProt Consortium. Ongoing and future developments at the universal protein resource. *Nucleic Acids Research 39* (2011), D214–D219. http://www.uniprot.org.

Apéndice A

Ejemplos de algunos formatos bioinformáticos

A.1. FASTA

```
>gi|24666374|ref|NM_079417.2| Drosophila melanogaster Gram-negative bacteria binding protein 2 (GNBP2)
ACAGTCCCACCGCAGAGCTTCTCAGATTAATACGCTAATCGGATTCTAGTTAGTACACATATTAAGAGTA
ATTAAAACTGCGTCATGAGGTGGGAATTTCTGCCATGTTTATTATTGCTCATTTCAAACAATAAAATCTT
TGGTTTTAAAGTACCCAGTATTAATTTTGAAATGCTAAAGGATGAAGGATTCGAAGTGTCCATACCAGAT
GAGCCTGGCATACAGCGGGTGTTCTACATGTTCCAAATTGACGACACCTGCCCGGCTCTAATGGACTACA
TCACGGAGGCAGTGAACGGAAGTTGGGTCTCCAAGCAAAAGATGAGTCTGCAGAACAACGACAAGCTGCA
GATATCAATGCTGGTGCAGTTCAATGAGGAAATCTTCGAAAAGAGTGAAACCAGGGTGATCATAAACACC
CGGCTACTGACCACCAAAGACTCTAGCTCGCGAGGCATAACATTCCTTACAGGAGAGGGCGAGTGCCAGG
CATACCTAGCTCCTGCACAGCAAGCCAAACGCTGCAAGGCCGCCCAAACGATAGTGAGCAATGGACGCCA
TACTTGCCAGGGTGAACTGATCTTTGAGGACAACTTCTCGGAGGCGCAGCTGAACAAGACCACCTGGAAG
CATGACATCCGACAGCGCATGTACCACGTGGAGGAGGAGCTGGTGGCCTTCGACGATGCCGCACGCAACT
GCTTTGTGAAGGAAGGCGAACTCCATATCGTTCCCACTATCGCCACCGAGGTCACCGATGGCAGCTTCAA
ACTGGGAGATCGCTGTACGGCCGTCGAAAGTCCGGAACAGGAGTGCAACATTGCGCACGGCATCTTCTAT
AGCATCAAGCCTCCAGTTTTCTCCGCCCAAATTCACACCAGAAACTCTTTCAGTTTCAAATTTGGCAAAA
TCGTCGTGCGCGCCAAGTTGCCCAAGGGGGATTGGCTTTTCCCCTATCTGATGCTCCAACCGGTTTCCAC
TTACGCGGAGACCCACTATGCCAAGCAATTGAGAATTGCATATGCCCGTGGCAATGCCAATTTGAGGACG
AAACAAGGAGATGATATCTCCGGAAACCACTTATATGGCGGCGGTGTGGTTTGGCACCACGGCAATGCAG
TACAGTTCCTCAAGGACAAGATAAGCAACAGCCACTACGGGGACGACTTTCACAACTACACCATGATCTG
GCAGAGGGATAAGATTACCCTAATGGTAGACGATGAGGTTTATGGGGAGCTGTACGACGGACTGCCATTC
TTCAACGAGAAGTGCTTTATTATCTTTGGCGTTACGGTGGGCGGCTTCCTGAACTTCGACGATAGTCTAT
TGGCCAAGGATGTCAAGCCTTATAAAAACAGGGAACCCCGGGCTGCCCTCTCATTTTGGCAGCACCGCGA
TGCTTGGGCCCCCACTTGGGGCAGGCACAGCGCCATGGTCATAGACTATGTTAGGGTTTATGCGGAGTAG
CTGTGGCATTTGTCACCAAAATATAAATAAAGTACTTAAACATCA
>gi|24666389|ref|NM_079418.2| Drosophila melanogaster Gram-negative bacteria binding protein 1 (GNBP1)
ATCGCAGCCGGAGCTGATATCAACGTGGATTTGCATTCGCCGAATCGCGATTGTCCGGACCAACTTAAAG
TAAACAAATAATAGTAATCGCTCCTCACCAGCGACAGAAACCGGTCTGCCAGTGAGATAGACACACAGAT
TGAGTCAGAGGCTCGGGATTCTTGTCGAAAAGTGCTGAACTCGAGGATTCAAAATGCCAGGATTGTGCAT
TGGAATCCTTCTGCTCATTGGTTTTGGATGCACCACTGCCTACAAGATCCCCACACCGACTGTGGAGCTC
CTTGAGACTGGATTTAGCGTTTCCATACCGGATGAAGAGGGCGTAAAAGTGGTGGCCTTTAATGTAAATC
GCAATCGGAATTTCACATCCTTCATAAACGAGGGACAGTACAATGTGAGATTGACTGAACCCCAGAACGG
CAGGTGGACGACCAACTTCAGTTCGGTTCCCTTGAGATCCCAAGATGTTCTATACCTTTGGACAAGTGTG
CAGCACCAAAAGGCTGTGTATCAAGATCTGGCGCAGCCCCTGCCAGTCTGCAATCTCGGCGGAGAGTATC
GGCCCAGGGGATGTTCGCCCGGTGATGATGACTTTACGGATGACAACCAGCTAAGTACTGAGGACAGTGC
TTTGGAACCCACCGCTCCCTCCGTCTGTGAACCTTCTGAAAGCCAGGTTTCGCCGCAAATCGGTGTTTCC
ATATGTAAGGGACAACTTTTGTTTGAGGAGACTTTTGATCAGCTGAATGAATCTCTGTGGATACATGATG
TTCGCCTGCCCCTCGACTCCAAGGATGCAGAGTTCGTCTTGTACGACGGGAAGGCCAAAGTTCACGATGG
CAACCTGGTGATTGAGCCGCTTCTTTGGTCCAGCTATCGCCCGGATCTCTCCATAGCCAACTCCAGGCTC
GATCTATCGGAGCGTTGTACTGGAACCCATAACAGAATTAAGGAATGCATCCTGCATTCCACGGGCAGTG
GACCCAGTGGGATAATGCCCCCAATTGTAACACCTCGAATCAGCACCAAGGAAACCTTTGCCTTTCAATA
```

```
CGGACGCATTGAGATCCGTGCCAAGCTACCAAAGGGAGATTGGATAGTACCACTCCTACTACTCGAACCC
CTCACCGAATGGTACGGGCAATCGGGCTACGAATCCGGCCAGCTGCGAGTGGCCTTGGCCAGGGGTAACT
CCGTGCTGAGGATGCCGCGCGGAAAGCTTGTAGATGGTAGATCCCTATACGGAGGACCCGTACTCTCCAC
GGATGCCCACCAAAGGGAGGACTTATGGCTGAGCAAGCGAAAGATATCCCATTTCGGCGATGACTTTCAC
ACCTACAGCTTGGATTGGAGCAGCAATAGGCTGCTCTTCTCCGTAGATGGTCAGGTCTATGGAGAGATGC
TGAATGGTTTCACTGAACTGGACGAGAATCCCAGGTGGAAGCAGGGCGGTCCCATGGCCCCGTTCGATAA
AATGTTTTACATTTCGTTGGGCGTCTCTGTGGGCGGATTTGGTGACTTTGTGGACCATCTTCGCACCGCC
ACTTATGAGAAGCCTTGGGCCAACTATCATCCCCAGGCGAAGCTGCAGTTCCACCAGGCCCAAGACCAGT
GGCTACCCACATGGAAACAGCCGGCCTTGAAGATCGATTATGTTCGTGTCTTCGCCAACTGATAACTAAA
ATCCATGATTTAGGTAGATGTTTATAAAATTACACATAAGTGAACGTGCCGATCGAAATCTATAAATCTA
TAAATAAATAAAGTCCTAAGAGAAACTCA
```

A.2. BLAST

```
BLASTX 2.2.21 [Jun-14-2009]


Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs",  Nucleic Acids Res. 25:3389-3402.

Query= anasbgenitals_2010-09-30_c10091
         (765 letters)

Database: ../SB_ANNOTATION+/ALL_PEPTIDES_ANNO.fa
           20,460 sequences; 11,306,574 total letters

Searching.....................................................done



                                                                  Score    E
Sequences producing significant alignments:                       (bits) Value

DLA_XX_002770|NP_001116739.1|beta-lactamase-like  |LG20|7053449|...   386   e-113
DLA_X_008800|LACTB2|lactamase -- beta 2 |LG10|22401574|22404911|...    44   9e-10
DLA_GroupXX_001250|ETHE1|ethylmalonic encephalopathy 1 |GroupXX|...    41   6e-09
DLA_GroupXX_001240|ETHE1|ethylmalonic encephalopathy 1 |GroupXX|...    38   5e-08

>DLA_XX_002770|NP_001116739.1|beta-lactamase-like
           |LG20|7053449|7055959|+|ENSDARP00000077664
          Length = 276

 Score =  386 bits (991), Expect = e-113
 Identities = 184/188 (97%), Positives = 185/188 (98%)
 Frame = +1

Query: 151 DWYAHKSLGDGLFWIQERFYQSDNRANIWLLRGSHQDVVIDTGLGLRSLPDYIDGKGLLG 330
           DWYAHKSLGDGLFWIQERFYQSDNRANIWLLRGSHQDVVIDTGLGLRSLPDYIDGKGLLG
Sbjct: 5   DWYAHKSLGDGLFWIQERFYQSDNRANIWLLRGSHQDVVIDTGLGLRSLPDYIDGKGLLG 64

Query: 331 KDPQRKNPLLAIATHAHFDHSGGLHQFQQVGVHSAEVDALANGDNFETVTWLSDREIAEA 510
           KDPQRKNPLLAIATHAHFDHSGGLHQFQQVGVHSAEVDALANGDNFETVTWLSDREIAEA
Sbjct: 65  KDPQRKNPLLAIATHAHFDHSGGLHQFQQVGVHSAEVDALANGDNFETVTWLSDREIAEA 124

Query: 511 PSPGWRARHYKGSRGVQPTHILQEGDVINLGDRQLTVLHMPGHSRGSICLHDHDNKLLFS 690
           PSPGWRARHYK  + VQPTHILQEGDVINLGDRQLTVLHMPGHSRGSICLHDHDNKLLFS
Sbjct: 125 PSPGWRARHYK-VKAVQPTHILQEGDVINLGDRQLTVLHMPGHSRGSICLHDHDNKLLFS 183

Query: 691 GDVVYDGS 714
           GDVVYDGS
Sbjct: 184 GDVVYDGS 191
```

```
>DLA_X_008800|LACTB2|lactamase -- beta 2
          |LG10|22401574|22404911|-|ENSGACP00000023205
         Length = 287

 Score = 43.5 bits (101), Expect = 9e-10
 Identities = 42/160 (26%), Positives = 71/160 (44%), Gaps = 2/160 (1%)
 Frame = +1

Query: 229 NIWLLRGSHQDVVIDTGLGLRSLPDYIDGKGLLGKDPQRKNPLLA--IATHAHFDHSGGL 402
           N +L+    + V+IDTG     +P+YI     L +   + N  +   I TH H DH+GG+
Sbjct: 32  NTYLVGTGERRVLIDTGEP--GVPEYISS---LKQALSQFNTSIQEIIVTHWHHDHTGGV 86

Query: 403 HQFQQVGVHSAEVDALANGDNFETVTWLSDREIAEAPSPGWRARHYKGSRGVQPTHILQE 582
                           D    +T  S+  +++ P    R+   + + G +    L++
Sbjct: 87  E------------------DICRDITG-SEVRVSKLP----RSSQVRETAGNKSFTFLKD 123

Query: 583 GDVINLGDRQLTVLHMPGHSRGSICLHDHDNKLLFSGDVV 702
           GDV+      L VL  PGH+   + L   +++ LFSGD +
Sbjct: 124 GDVVQTEGATLKVLFTPGHTDDHMALLLEEDQALFSGDCI 163


>DLA_GroupXX_001250|ETHE1|ethylmalonic encephalopathy 1
          |GroupXX|4137938|4140523|-|ENSTNIP00000007684
         Length = 248

 Score = 40.8 bits (94), Expect = 6e-09
 Identities = 36/120 (30%), Positives = 48/120 (40%), Gaps = 2/120 (1%)
 Frame = +1

Query: 349 NPLLAIATHAHFDH--SGGLHQFQQVGVHSAEVDALANGDNFETVTWLSDREIAEAPSPG 522
           N  +A+ TH H DH  S GL + + VG+ SA             ++ LS
Sbjct: 67  NLKVAVNTHCHADHITSTGLMKKRLVGLKSA-------------ISKLS----------- 102

Query: 523 WRARHYKGSRGVQPTHILQEGDVINLGDRQLTVLHMPGHSRGSICLHDHDNKLLFSGDVV 702
                  G      L EGD IN G   LTV   PGH+ G + L   D  + F+GD +
Sbjct: 103 ----------GASADIHLSEGDKINFGKHYLTVRETPGHTDGCVTLVLGDQSMSFTGDAL 152


>DLA_GroupXX_001240|ETHE1|ethylmalonic encephalopathy 1
          |GroupXX|4132548|4135284|-|ENSGACP00000016597
         Length = 284

 Score = 37.7 bits (86), Expect = 5e-08
 Identities = 18/43 (41%), Positives = 23/43 (53%)
 Frame = +1

Query: 574 LQEGDVINLGDRQLTVLHMPGHSRGSICLHDHDNKLLFSGDVV 702
           L EGD I  G   LTV   PGH+ G + L   D  + F+GD +
Sbjct: 146 LSEGDKIPFGKHYLTVRETPGHTDGCVTLVLEDQSMAFTGDTL 188


  Database: ../SB_ANNOTATION+/ALL_PEPTIDES_ANNO.fa
    Posted date:  Dec 9, 2010  1:01 PM
  Number of letters in database: 11,306,574
  Number of sequences in database:  20,460

Lambda     K      H
   0.318    0.134    0.401

Gapped
Lambda     K      H
   0.267   0.0410    0.140
```

```
Matrix: BLOSUM62
Gap Penalties: Existence: 11, Extension: 1
Number of Sequences: 20460
Number of Hits to DB: 17,791,473
Number of extensions: 458420
Number of successful extensions: 1530
Number of sequences better than 1.0e-05: 4
Number of HSP's gapped: 1524
Number of HSP's successfully gapped: 4
Length of query: 255
Length of database: 40
Length adjustment: 0
Effective length of query: 255
Effective length of database: 40
Effective search space:    10200
Effective search space used:    10200
Neighboring words threshold: 12
Window for multiple hits: 40
X1: 16 ( 7.3 bits)
X2: 38 (14.6 bits)
X3: 64 (24.7 bits)
S1: 41 (21.7 bits)
S2: 33 (17.3 bits)
```

A.3. Pfam scan

```
Contig1590      GO:0046872      muscle lim protein
Contig1596      GO:0032561      tubulin beta-1 chain
Contig1596      GO:0043623      tubulin beta-1 chain
Contig1596      GO:0007017      tubulin beta-1 chain
Contig1596      GO:0017111      tubulin beta-1 chain
Contig1596      GO:0032991      tubulin beta-1 chain
Contig1596      GO:0015630      tubulin beta-1 chain
Contig1601      GO:0003824      fructose-bisphosphate aldolase a
Contig1609      GO:0043005      palmitoyl-protein thioesterase 1
Contig1609      GO:0000737      palmitoyl-protein thioesterase 1
Contig1609      GO:0007417      palmitoyl-protein thioesterase 1
Contig1609      GO:0005626      palmitoyl-protein thioesterase 1
Contig1609      GO:0016291      palmitoyl-protein thioesterase 1
Contig1609      GO:0006665      palmitoyl-protein thioesterase 1
Contig1609      GO:0007040      palmitoyl-protein thioesterase 1
Contig1609      GO:0000323      palmitoyl-protein thioesterase 1
Contig1609      GO:0005576      palmitoyl-protein thioesterase 1
Contig1609      GO:0048468      palmitoyl-protein thioesterase 1
Contig1609      GO:0050890      palmitoyl-protein thioesterase 1
Contig1609      GO:0007610      palmitoyl-protein thioesterase 1
Contig1609      GO:0044425      palmitoyl-protein thioesterase 1
Contig1609      GO:0007268      palmitoyl-protein thioesterase 1
Contig1609      GO:0006464      palmitoyl-protein thioesterase 1
```

A.4. Pfam scan

```
# pfam_scan.pl,  run at Wed Nov 24 18:44:13 2010
#
# Copyright (c) 2009 Genome Research Ltd
# Freely distributed under the GNU
# General Public License
#
# Authors: Jaina Mistry (jm14@sanger.ac.uk), John Tate (jt6@sanger.ac.uk),
#          Rob Finn (rdf@sanger.ac.uk)
#
```

```
# This is free software; you can redistribute it and/or modify it under
# the terms of the GNU General Public License as published by the Free Software
# Foundation; either version 2 of the License, or (at your option) any later version.
# This program is distributed in the hope that it will be useful, but WITHOUT
# ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
# FOR A PARTICULAR PURPOSE. See the GNU General Public License for more
# details.
#
# You should have received a copy of the GNU General Public License along with
# this program. If not, see <http://www.gnu.org/licenses/>.
# = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
#      query sequence file: /home/inb/projects/MolluSea/results/
20101024171418_97500_698735/tmp/Contig30.faa.nrdb100ncbi.frames
#        searching against: /home/inb/gridded-apps/PfamScan/data/Pfam-A.hmm, with cut off
--cut_ga
#    resolve clan overlaps: on
#     predict active sites: on
# = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
#
# <seq id> <alignment start> <alignment end> <envelope start> <envelope end> <hmm acc>
<hmm name> <type> <hmm start> <hmm end> <hmm length> <bit score> <E-value> <significance>
<clan> <predicted_active_site_residues>

Contig30_frame_3            18     77     16     78 PF05160.6   DSS1_SEM1         Family
2    62    63     64.9   3.2e-18   1 No_clan
#HMM       kkeekklkllEeDDEFEdFpvedweeeeeekeaskelWeedWDDddveddFskqLkeelek
#MATCH     k+++++l llEeDDEFE+Fp+edw++ +e++++  ++We++WDDd+v+dd s qL++elek
#PP        67889999**********************99997.************************98
#SEQ       KEKKPDLGLLEEDDEFEEFPAEDWNSADEDEKDI-NVWEDNWDDDNVDDDLSVQLRNELEK
```

Apéndice B

Entrevistas

B.1. Entrevista 1

NOMBRE: Juana Maria Navarro Llorens
FECHA: 15/03/2011
Institución: Facultad de Biología de la Universidad Complutense de Madrid

---

CONCLUSIONES

Filtros iniciales

- Buscar estructuras no codificantes y reguladoras ->encontrar zonas del genoma que son ARN reguladores, etc ->BLAST x ARN transferente
    - Son muy conservados dentro del genoma pero todavía no se sabe mucho de reguladores.
- Encontrar elementos de inserción y transponsones ->están muy conservados, no codifican proteínas
    - Pueden tener longitud de 200bp a 1000bp
    - Para encontrarlos: BLAST contra Base de Datos de transponsones

Pipeline básico

Análisis del Informe de BLAST

- Coger los 5-10 primeros *hits* con *e-value* $>= e^{-30}$
- Buscar la raíz del término que los define
    - Si hay diferentes raízes:
        - Saber el porcentaje que aparece.
        - Si el porcentaje es muy próximo entre 40-70 %, dar preferencia al dominio obtenido por el Pfam scan para definir raiz.
    - Si las raízes son "hypotetical" y no hay dominio
        - Coger los 5-10 siguientes y volver a buscar raíz del término
    - Si las raizes son 'hypotetical" y hay dominio
        - Coger el dominio y definir raíz
    - Else
        - Definir raíz

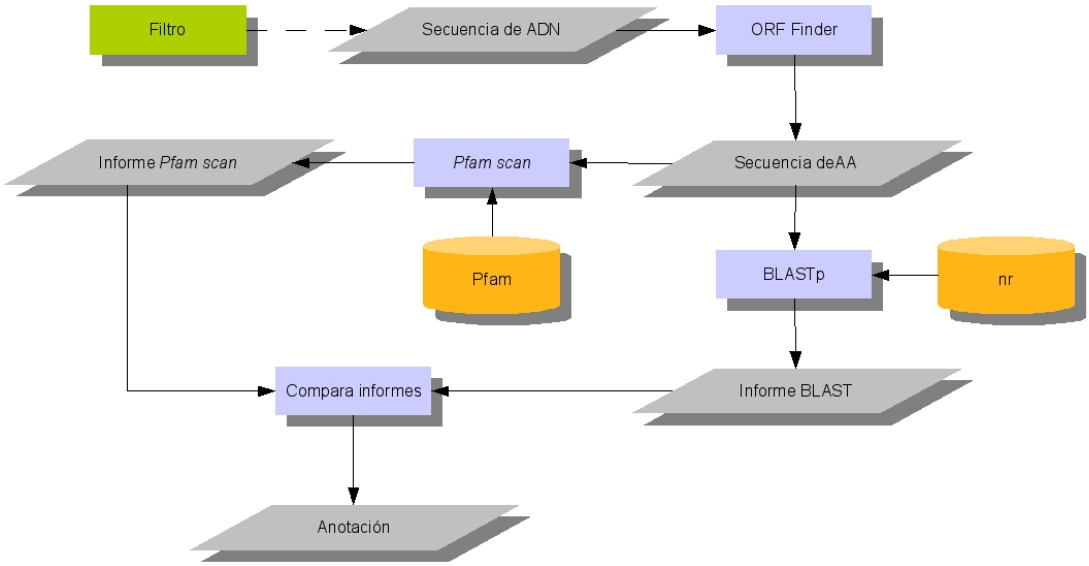

Figura B.1: *Esquema del pipeline descrito en la Entrevista B.1.*

- Cuando no existe consenso en cuanto a la longitud del *hit*, puede haber un péptido señal. En este caso, se queda con el *hit* mayor y se puede utilizar el SignalP pata buscar el péptido señal.
- A veces, el gen aparece "truncado", es decir, un trozo aparece en un marco de lectura mientras el otro aparece en otro marco.
    - Éstos son "genes" son pseudogenes y son anotados como proteína truncada
- La identidad debe ser de mínimo de 70 %, en general, excepto si:
    - Si es regulador: identidad mínima de 30 %
    - Si es enzima: identidad mínima de 60 %
    - Else se mira el dominio

Análisis del Informe de BLAST

- Hay proteínas que no poseen dominio o que su dominio no está catalogo o no es conocido
- Si existe un único dominio
    - Comparase este dominio con el resultado obtenido por el BLAST
        - Si iguales asignar dominio
        - Si identidad $< 70\,\%$ (sin estar hacer parte de algunas de las exceptos descritas anteriormente) asignar el dominio obtenido
        - Si diferentes investigar más (artículos o otros programas)
- Else
    - Se cogen los dominios con el *e-value* menor y que no solapan para comparar con los resultados del BLAST.

B.2. Entrevista 2

NOMBRE: Javier Tamames
FECHA: 24/03/2011
Institución: Centro Nacional de Biología (CNB)

CONCLUSIONES

- BLAST X PROSITE (BD de perfiles)
  - If *hit*
    - Coger los 10-20 primeros
    - Buscar la familia más común de los *hits*
    - Coger las secuencias de las proteínas pertenecientes a cada *hit* y hacer un alineamiento múltiple con la secuencia del *query*
    - Buscar evidencias de que el *query* está tan conservado como las proteínas del alineamiento: zonas importantes, como el centro activo
      - Se no hay información suficiente para tomar una decisión, buscar más datos en artículos
  - else
    - Buscar más información

¡Consejo! Para analizar los resultados del BLAST normalizar todos los hits con relación al *bitscore* y utilizar este valor en lugar del *e-value*.

B.3. Entrevista 3

NOMBRE: José Manuel Rodriguez Carrasco
FECHA: 25/03/2011
LOCAL: Instituto Nacional de Bioinformática (INB)

CONCLUSIONES

- BLASTx contra nr (corte de e-value: -e 0.000001)
  - If *hit*
    - Salida xml ->BLAST2GO
  - else
    - Traducir secuencia en los seis marcos de lectura y ejecutar *Pfam scan* para cada marco.
- BLASTx contra Uniprot (corte de e-value: -e 0.000001)

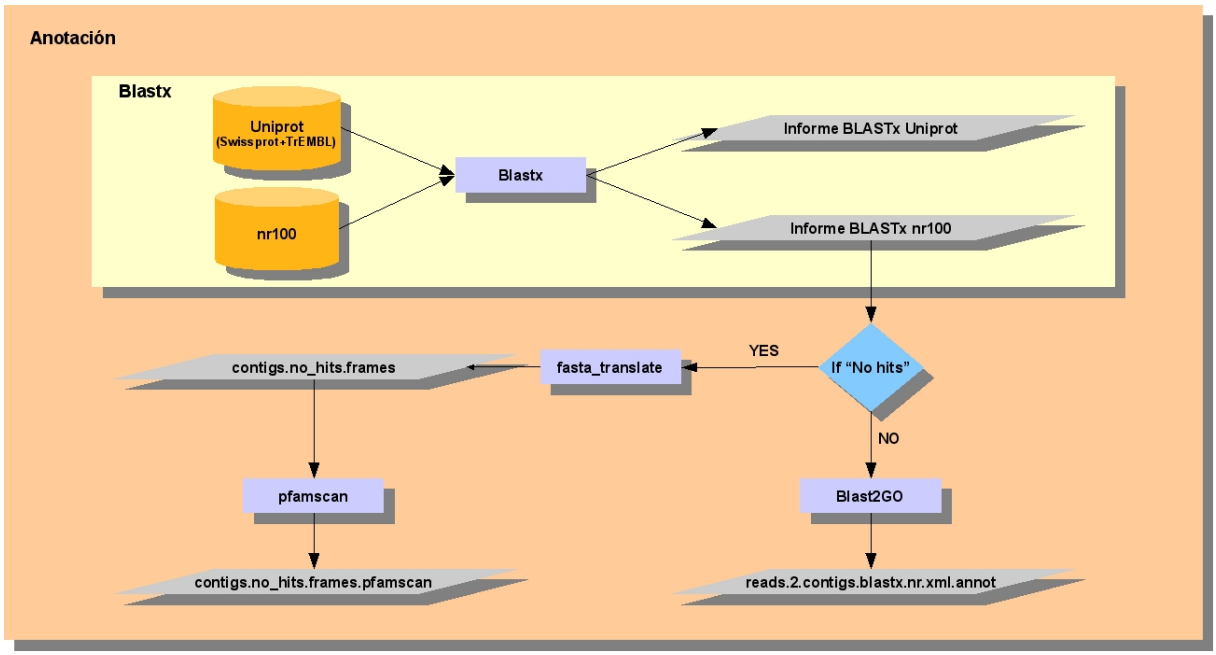

Figura B.2: *Esquema del pipeline descrito en la Entrevista B.3.*

Apéndice C

Resultados de la implementación

C.1. Resultados

Se ha implementado el esquema propuesto en la Figura 4.5 en la forma en que fue descrito anteriormente en la Sección 4.7, habiendo sido probado con las secuencias de *Venerupis philippinarum* de la Base de Datos de vertebrados marinos MolluSEA, disponible en `http://aquasea.ucm.es/mollusea`. La anotación de estas secuencias es clasificada como **calidad 1**, cuando la anotación asignada tiene elevado grado de fiabilidad, comprobado mediante la existencia de sitios activos conservados en la secuencia analizada. No obstante, la anotación es clasificada como **calidad 2** cuando es fiable, pero no existe información sobre las coordinadas de los sitios activos. En el caso de que los datos utilizados para anotar las secuencias no superen el umbral estipulado en la Base de Conocimiento no se asigna ninguna anotación a la secuencia, evitando así anotaciones erróneas.

Como muestra la gráfica de la Figura C.1, del total de 888 secuencias de ADN asociadas a *Venerupis philippinarum*, 62 no han obtenido ninguna información al ejecutar *Pfam scan* o BLAST, es decir, no pueden ser anotadas a través de las Bases de Datos utilizadas. Se ha adquirido información de 826 secuencias, donde 143 fueron anotadas con calidad 1, mientras 112 tienen calidad 2 y 571 no fueron anotadas.

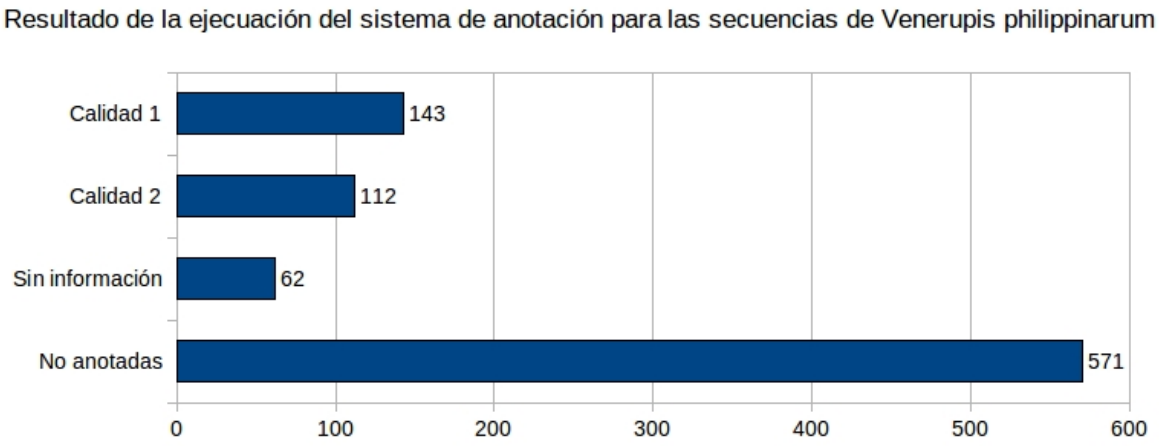

Figura C.1: *Resultados obtenidos de la ejecución del Sistema Experto desarrollado, con base en el esquema de la Figura 4.5, para las secuencias de Venerupis philippinarum.*

A pesar de que el sistema desarrollado ha anotado solamente 29 % de las secuencias, este es un resultado es bastante satisfactorio, puesto que las reglas empleadas para inferir las anotaciones son muy estrictas, lo que aumenta el grado de fiabilidad de los resultados obtenidos. Además, existen secuencias que no pueden ser anotadas con la información disponible en las Bases de Datos públicas, ya que estas aún no han sido catalogadas. Una alternativa para intentar anotar las secuencias que no han sido anotadas con el sistema desarrollado es añadir otras Bases de Datos al sistema, lo que puede ser fácilmente realizado debido a su modularidad.